The Dissertation Committee for Wei Dong
certifies that this is the approved version of the following dissertation:

Collaborative Mobile Services

Committee:

Yin Zhang, Supervisor

Lili Qiu, Supervisor

Aloysius K. Mok

Andrew Whinston

K. K. Ramakrishnan

Collaborative Mobile Services

by

Wei Dong, B.E., M.S.C.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2013

Collaborative Mobile Services

Wei Dong, Ph.D.
The University of Texas at Austin, 2013

Supervisors: Yin Zhang
Lili Qiu

Mobile devices like smartphones and tablets are being adopted with unprecedented speed. The growth in demand and system complexity increasingly requires collaboration of multiple parties in order to achieve better functionality, efficiency, performance, etc. This poses unique challenges such as information sharing among different parties, utility sharing among different parties, and dishonest and collusive behaviors.

Different mobile services may require different types of collaboration and involve different entities in the system. In this work we take a bottom-up approach by first looking at collaboration at the end user level, then the cross level collaboration and finally at the service provider level. Specifically, we first consider a completely distributed service: friend discovery in mobile social networks, where users of a mobile social network work together with each other to discover potential new friends nearby by computing their social proximity. We develop mathematically sound yet highly efficient approaches

that simultaneously achieve privacy and verifiability. We then focus on cellular offloading where a cellular service provider seeks third party resource to offload cellular demand, as an example of cross level collaboration. We propose a reverse auction framework: iDEAL, which efficiently allocates cellular resource and third party resource in a joint optimization, effectively incentivize third party resource owners and mitigates dishonest and collusive behaviors. We validate our findings and approaches with real trace driven analysis and simulation, as well as real implementation. Finally we focus on collaboration at the service provider level and propose a double auction framework - DA^2. DA^2 allows cellular service providers to reallocate spectrum resource in a dynamic fashsion. It preserves all the desired economic properties. Compared with existing spectrum double auctions, DA^2 achieves higher efficiency, revenue, and spectrum resource utilization, due to its ability to more accurately capture the competition among buyers, which is characterized by a complex conflict graph. We evaluate DA^2 and demonstrate its superior performance via simulations on conflict graphs generated with real cell tower locations.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Motivation: As the world embrace new generation of mobile devices such as smartphones and tablets, many different mobile services start to emerge and become part of people's everyday lives. As the demand and system complexity keeps growing, it's becoming increasingly important that many services needs collaboration of multiple parties in order to ensure usability, achieve high efficiency and performance. For such collaboration to be effective, several unique challenges need to be addressed:

- Information sharing among different parties. Information sharing is critical for many collaborations to be effective but is hard to achieve. Most companies consider information as an important asset and individual users are also becoming increasingly aware of their privacy. It's unclear how to avoid unnecessary information leak and in the meantime facilitate efective collaboration.

- Utility sharing among different parties. In order to enable certain collaboration, each party must be given certain incentives. While incentives have been well studied in environments such as P2P, the mobile environment is very different. Moreover incentive is also service dependent.

The form of incentive, the amount of incentive, the party that gives the incentive and the party that receives the incentive can vary a lot when different services are considered.

- Dishonest and collusive behaviors. When multiple parties are involved, it's hard to assume trust between all parties, especially when one party can lie or a few parties can collude to maximize their utility. It's critical to defend against such behaviors such that the collaboration is fair and sustainable.

There are a wide spectrum of mobile services. Some are closer to the service provider while others may be closer to end users. Collaboration can happen within each level or even across levels. To develop general principles and techniques for enabling collaborative mobile services, in this work we take a bottom-up approach, where we first consider a mostly distributed service that only requires collaboration between end users: friend discovery in mobile social network, and then we study the cross level collaboration between a cellular service provider and third party resource owners who are able to offload cellular traffic, finally we focus on collaboration between cellular service providers by enabling spectrum resource allocation in a dynamic and flexible fashion.

Friend discovery in mobile social networks: An essential capability offered by mobile social networks is to allow mobile users to discover and interact with friends who happen to be in their physical vicinity. Suppose you are waiting for your flight in an airport and your mobile phone discovers your friend's

friend is in the next aisle and you can talk with face-to-face. Or you visit a new place and your mobile phone finds someone in your vicinity shares similar attributes as you so that you can interact with. Though promising, it creates serious privacy and security concerns. In particular, people are often reluctant to reveal their presence and personal profile to an arbitrary person in their vicinity. It is also unwise to blindly trust information received from an arbitrary person. While similar issues also exist in online social networks, these concerns become more serious because mobile social networks blur the boundary between the cyberspace and the physical world. Moreover, the broadcast nature of wireless medium also makes it easy for a malicious user to spoof and inject traffic into the mobile social networks.

This is an important yet challenging problem because it involves joint computation between two parties that do not trust each other. The standard approach for predicting friendship is based on the notion of *proximity measure*, which quantifies the closeness or similarity between nodes in a social network. Many proximity measures have been proposed, including the number of common neighbors, the number of common attributes, cosine similarity, and path-ensemble-based measures. As shown in Section 2.2, they can all be cast as a dot product operation on two vectors. Therefore we consider proximity computation as a dot product operation without loss of generality. To identify potential friends, two users can exchange their social coordinates and compute the proximity between them; if their proximity exceeds a threshold, they try to make friends with each other.

However, directly exchanging social coordinates and computing their dot product opens door to a variety of attacks. In particular, we identify two serious attacks: (i) fingerprinting an individual user (based on either her social coordinate or her proximity with another known social coordinate), and (ii) falsifying proximity (*e.g.*, an attacker forges a social coordinate close to a target user's social coordinate to trick the user to make friend with the attacker). Defending against these attacks is particularly challenging because we want to simultaneously achieve two conflicting goals: ensuring verifiability (so that a malicious user cannot forge his/her social coordinate or forge the outcome of proximity computation), yet preserving privacy (*i.e.*, divulge no private information if the true proximity between two users is below the desired threshold).

To address the challenge, we develop novel techniques and protocols for computing proximity in a privacy-preserving, verifiable, and efficient manner. Specifically, we first develop a proximity pre-filtering protocol for determining whether the proximity between two users exceeds a given threshold. The protocol ensures that the initiator can only learn the comparison result between the estimated proximity and the threshold.

The protocol does not involve any expensive cryptographic computation and is thus highly efficient. However in this protocol a malicious user can forge the comparison result. To defend against such attacks, we then develop two secure dot product protocols: one is based on homomorphic cryptography and the other leverages both homomorphic cryptography and obfuscation for

higher efficiency. To the best of our knowledge, they are the first secure dot product protocols that are both privacy-preserving and verifiable.

We demonstrate the effectiveness of our protocols using both analysis and real implementation on smartphone.

Incentivize cellular offloading via auctions: The explosive growth of cellular traffic and its highly dynamic nature make it increasingly expensive for a cellular service provider to provision enough cellular resources to support all its consumers at all times. The current best practice is for service providers to augment the cellular network capacity by deploying alternative wireless technologies (*e.g.*, Wi-Fi and femtocells) on their own. While this approach is helpful in alleviating the stress on the busiest cellular regions in the short term, it alone is not sufficient in the long run due to the high deployment cost and excessive interference.

A better alternative is to leverage resources *on demand* from third-party resource owners by buying capacity whenever needed. On-demand purchase of such resources can potentially lead to a win-win solution: the cellular service provider achieves significant savings by not having to provision for the peak traffic demands; the third-party resource owners gain additional revenue from the otherwise wasted spare capacity; the overall user experience is also improved.

In order for the auction based cellular offloading to be successful, however, it is essential to have an incentive framework that can effectively fos-

ter collaboration while guarding against non-truthful and collusive behavior. Specifically we identify the following unique challenges, none of which have been considered earlier.

- *Diverse spatial coverage.* Cellular resources can serve traffic anywhere in a cell sector (albeit at different rates depending on path loss etc.), whereas Wi-Fi hotspots and femtocells have a much more limited communication range, making it essential to consider the spatial coverage of different resources. However, one cannot simply partition resources into separate regions and launch independent reverse auctions within each region, because the longer-range cellular resource introduces coupling between the Wi-Fi hotspots or femtocells in different regions. For example, buying more resources from a cheaper Wi-Fi hotspot in one region frees up more cellular resources, which reduces the amount of cellular traffic to be offloaded from regions with more expensive Wi-Fi hotspots.
- *Traffic uncertainty.* Cellular traffic is highly dynamic and unpredictable. Since the cellular service provider has to purchase third-party resources based on predicted traffic demands at a future time, it can easily result in under-provisioning or over-provisioning without an effective technique to cope with traffic uncertainties. In contrast, in conventional reverse auction settings, the total amount of goods that the buyer wants is typically known *a priori*.

- *Non-truthful bidding and collusion.* It is essential for us to explicitly guard against both non-truthful bidding and collusion. Due to the distributed nature of hotspot locations, collusion in our context is quite different from what were studied previously and calls for a new study to understand possible collusion strategies and mitigate them.

To address these challenges we propose iDEAL, a novel auction-based incentive framework to enable dynamic offloading of cellular traffic. In iDEAL, a cellular service provider purchases bandwidth on demand from third-party resource owners through *reverse auctions*, where a third-party resource owner may be a Wi-Fi hotspot owner, a femtocell owner, or another cellular service provider. In each reverse auction, the goods of interest are bandwidth resources, the bidders (*i.e.*, sellers or auctionees) are the third-party resource owners, and the auctioneer is the cellular service provider (or a trusted third party). Each bidder (*i.e.*, third-party resource owner) submits a bid that specifies the total amount of bandwidth it offers in the next time interval and the unit price the bidder asks for. After collecting all the bids, the cellular service provider determines (i) an *allocation*, *i.e.*, how to allocate its customer traffic between different third-party resource owners (depending on the region they cover) and its own cellular network, and (ii) a *price*, *i.e.*, how much it pays each third-party resource owner that offloads cellular traffic.

iDEAL has the following distinctive features: (i) iDEAL explicitly accounts for the spatial coverage of different resources and can effectively foster competition among third-party resource owners in different regions, resulting

in significant savings to the cellular service provider. (ii) iDEAL incentivizes bidders (*i.e.*, third-party resource owners) to participate in the reverse auction and to be truthful in their bidding. (iii) iDEAL is provably efficient in that the winners are the bidders who have the lowest valuation of their resources. (iv) iDEAL can effectively mitigate collusion. (v) iDEAL can effectively cope with the highly dynamic nature of traffic demands.

We present useful extensions to iDEAL: (i) support general bidding curves, which gives a hotspot owner the flexibility to submit its ask price in the form of a curve, such that different unit price is used when different amount of capacity is sold; (ii) support femtocell offloading and dynamic roaming; (iii) incorporate quality of service consideration (in addition to cost); (iv) potentially delay demands that are delay-tolerant to further reduce cost and improve efficiency, and (v) determine which users' traffic to offload.

We extensively evaluate iDEAL using simulation based on real traces from one of the largest US cellular service providers. Our results clearly demonstrate the effectiveness of our approach. We further demonstrate the feasibility of our approach using a simple prototype implementation.

Dynamic Spectrum Allocation via Double Auctions: As the world embraces wireless and mobile technologies at an unprecedented rate, wireless traffic is growing exponentially. Wireless traffic is also known to fluctuate heavily over time, following strong diurnal and weekly patterns [96, 72]. This explosive growth of wireless traffic and its highly dynamic nature can make it costly for a single wireless service provider to buy enough spectrum based on

a long-term contract so as to sustain its peak load, which may only last for brief periods. This inefficiency of long-term spectrum allocation, as prevalent today, motivates the need for dynamic spectrum access — wherein a wireless service provider only obtains sufficient spectrum to support the "typical" traffic demands and can (i) purchase additional spectrum on-demand from other providers to satisfy higher traffic demands, or (ii) offer the spare spectrum to other providers for profit when there is lower traffic demand.

While dynamic spectrum access is attractive and it is now technically feasible to dynamically change spectrum on-the-fly [35, 78, 93], an important open issue remains how to share spectrum across multiple parties. It is essential to have an incentive framework that can effectively foster collaboration while guarding against dishonest behavior.

We aim to achieve the following properties for the double spectrum auction, where the first three properties are necessary economic properties for a good double auction and the remaining three quantify the effectiveness of the auction: (i) *Truthfulness:* Bidders cannot benefit from bidding differently from their true valuation; (ii) *Individual rationality:* Bidders get non-negative utilities, *i.e.*, sellers are paid no less than their asks and buyers do not pay more than their bids; (iii) *Budget-balance:* The total amount paid to the sellers is no more than the total amount received from the buyers. This prevents the auctioneer, which runs the auction, from losing money; (iv) *Efficiency:* It is the difference between the sum of the winning buyers' valuations and the sum of the winning sellers' valuations. To achieve good efficiency, the goods should

be sold to the buyers that value them the most and be sold by the sellers that value them the least; (v) *Revenue:* It is the total amount of payment from all winning buyers. A winning buyer may pay different amount from its bid, depending on the auction design. A higher revenue gives sellers stronger incentive to participate. (vi) *Utilization:* Unique to spectrum auctions, we seek to maximize spectrum resource utilization by allowing as many buyers as possible to reuse the same spectrum resources. We quantify the utilization by the total number of buyers that are assigned spectrum.

Designing a good double auction for spectrum reuse poses the following significant challenges. (i) How to accurately capture wireless interference among the buyers, which is necessary to support spectrum reuse. Achieving spectrum reuse, where multiple non-competing buyers use the same resource simultaneously, is the key property that makes spectrum auction fundamentally different from traditional auction, which assumes all the entities compete against each other for an item. (ii) How to design a truthful double auction. Simply applying truthful auctions for the sellers and buyers does not lead to a truthful double auction. It is also critical to ensure participants cannot gain by manipulating the interaction between two sides of the market. It is especially challenging in a spectrum auction because the two sides of the market are very different. (iii) How to maximize the spectrum utilization while achieving budget balance. To maximize spectrum utilization, we need to sell as many channels as possible and let them be concurrently used by as many buyers as possible. However, this goal conflicts with the budget balance, since sell-

ing more channels means a lower average price and a possibly lower revenue (as the price is determined by the loser's price). A lower revenue means that fewer channels may be sold in a double auction due to the need to achieve budget balance. We will further elaborate these challenges using examples in Section 8.2.

The advantages of double sided spectrum auctions have attracted lots of research attention. Despite considerable previous work, significant challenges remain. In particular, some works fail to support spectrum reuse while preserving truthfulness [83, 90]. The first work that satisfies all economic properties and supports spectrum reuse is TRUST [95]. It follows the classic McAfee's double auction [54], which jointly computes the auction result for buyers and sellers. However, to apply the classic design, it makes several simplifications that could sacrifice performance. Specifically, it randomly groups non-interfering buyers and requires buyers in the same group to win or lose together. So the fate of a group is determined by the lowest bidding buyer. A group can lose even if it has many high bidding buyers, which results in unfairness and low efficiency. It also enforces uniform pricing for all buyers that win the same channel, in which case the price can be no more than the lowest bid of these buyers. Thus the total reveue is limited and that further hurts the auction performance because few channels can be traded while preserving budget balance. Several follow-up works share similar problems [91, 25].

We develop a novel double auction for dynamic allocation of spectrum (DA^2) in Section 9. DA^2 uses separate designs of the buyer and seller side

auctions while achieving truthfulness, budget balance, and individual rationality. Compared to the classic joint design strategy [54] taken by most existing spectrum double auction designs [95, 91], a unique advantage of our separate auction designs for the buyers and sellers is that this enables flexible combinations of different buyer/seller side designs. This significantly increases the design space and can immediately benefit from future enhancement to auction design on either buyer or seller side. Moreover, it also allows different properties of both sides to be captured accurately, which is especially important for spectrum double auctions since the buyer side is much more complicated due to wireless interference. We show how to combine the design of two sides to ensure budget balance and identify the necessary properties of a single side design in order for the double auction to be truthful.

DA^2 consists of three components: (i) seller side auction design, assuing N channels sell, (ii) buyer side auction design, assuming N channels sell, and (iii) determining N (*i.e.*, the number of channels) to sell to satisfy budget balance.

Among them, the buyer side design is the most challenging component due to complicated wireless interference relationships among the buyers. We propose to partition the graph into subgraphs based on the competition patterns, which gives us the flexibility to compute allocation independently in each subgraph and then combine the results. In particular, in order to support frequency reuse, buyers are grouped into independent sets, where buyers in the same independent set do not interfere. The performance of an auction is sensi-

tive to the independent set construction. However, the independent sets should be constructed in a bid-independent fashion to ensure truthfulness [95]. Graph partition reduces the randomness of independent set construction, and allows us to better capture the different competition patterns in each subgraph to achieve higher revenue by computing pricing in each subgraph independently.

We identify important requirements for a good partitioning algorithm: cutting as few edges as possible while yielding balanced partitions. These two requirements correspond to the well known metric, called the Normalized Cut. We apply spectral clustering [60], which is an effective approximation algorithm to minimize the Normalized Cut. We then use the group bid definition in [91] to sort the groups and determine potential winners in a subgraph. To combine the allocation results from subgraphs while preserving truthfulness, we develop a novel pairwise merge procedure to merge the results from two subgraphs at a time, while reducing the efficiency and revenue loss due to conflicts between subgraphs.

We further extend DA^2 to support localized sellers (*i.e.*, sellers that sell spectrum that can be used by buyers in certain regions) in Section 9.4. To optimize performance and maximize the number of trades to carry out, we develop *cross-region budget balance* to allow regions to help each other and let a seller win in all regions as long as it is budget balanced in any one region. Our new allocation algorithm naturally gives a seller's critical value (*i.e.*, the highest price that it can ask and still win) and we use it as payment to the seller.

We extensively evaluate our approaches using conflict graphs generated from real cell tower locations derived from a major US cellular provider (Section 10). Our results show that DA^2 consistently yields high efficiency, revenue, and utilization, and significantly out-performs the existing approaches.

Chapter 2

Friend Discovery in Mobile Social Networks

2.1 Overview

We consider how to identify potential friends in one's physical vicinity, which is essential to mobile social network. We assume that users only have occasional connectivity with a trusted server on the Internet (*e.g.*, once a day) and immediate communication occurs locally within a mobile social network and does not need access to the server; moreover the server does not know users' locations.

One way to address the privacy and security issues is to take advantage of a trusted central server, which collects information from individual users, computes and disseminates the results. Server-based solution is not suitable for mobile social networks for the following reasons. First, users in a mobile social network may not have direct access to a computer or the Internet. While cellular data service increases in popularity, the number of data service subscribers is still very limited due to its high cost [58, 87]. As a result, a server-based solution cannot work in such an environment unless dedicated servers are deployed to support mobile social networks, which is prohibitively expensive if not infeasible. In contrast, a distributed solution is more appeal-

ing because it obviates the need of always having access to a server. Second, as people are increasingly concerned about their location privacy and personal data, they may not want to reveal their current location or other personal information even to a trusted server. Wills *et al.* [59, 45] studied 13 mobile online social networks, such as Facebook, Friendster, Hi5, LinkedIn, MySpace, Twitter, and found all of them leaked some private information to tracking sites and several of them passed users' location information to a third party. Third, the server can easily become a bottleneck, a single point of failure, and the target for denial-of-service attack. These limitations of server-based approach motivate us to move away from a centralized solution and design distributed secure protocols for friend discovery in mobile social networks.

Before we introduce our solution, in this chapter we first provide some background on coordinate-based proximity estimation. Then we identify potential attacks against friend discoveery, using real social network data.

2.2 Coordinate-based Proximity Estimation

In this section, we first introduce the notion of proximity measure and describe how one can compute social proximity based on social coordinate. Our key assumption is that users are already part of a common (online or physical) social network. We can then identify potential friends by checking whether two users are sufficiently close in this social network.

Proximity measure in social networks. A *social network* [85] is a so-

cial structure modeled as a graph, where nodes represent people and edges represent relationships between them (*e.g.*, friendship). A central concept in social networks is *proximity measure*, which quantifies the closeness or similarity between nodes in a social network. Proximity measure serves as the basis for many social network applications (*e.g.*, [34, 14, 28, 92]). As a result, a variety of proximity measures have been proposed. The simplest proximity measures include the number of common neighbors or the number of common attributes between the two users. More sophisticated proximity measures involve infinite sums over the ensemble of all paths between two nodes in the social networks (*e.g.*, Katz measure [42], rooted PageRank [49, 50], and escape probability [79]). Compared with simple proximity measures, path-ensemble based proximity measures capture more information about the underlying social structure and have been shown to be more effective in social networks [49, 50, 79]. In particular, the Katz measure is shown to be particularly effective in predicting new links in social networks [49, 50, 76]. It is defined as

$$\mathsf{Katz}[x, y] = \sum_{\ell=1}^{\infty} \beta_{\text{Katz}}^{\ell} \cdot |\mathsf{paths}_{x,y}^{\langle \ell \rangle}| \tag{2.2.1}$$

where $\mathsf{paths}_{x,y}^{\langle \ell \rangle}$ is the set of length-ℓ paths from x to y in a social network, and $\beta_{\text{Katz}} < 1$ is a damping factor. We focus on computing Katz measure in our evaluation, but our protocols are general and can be applied to many other proximity measures including those simple proximity measures, since proximity computation can be considered as a dot product operation (also known as the inner product) without loss of generality (as shown below).

Estimating proximity from social coordinates. [76] developed a technique called *proximity embedding* for efficient and accurate computation of path-ensemble based proximity measure (*e.g.*, Katz measure, rooted PageRank, escape probability) in large social networks with millions of nodes. It approximates the entire $m \times m$ proximity matrix P, where $P(i, j)$ denotes the proximity between users i and j, as the product of two rank-r factor matrices U and V, where m can be millions but r is much smaller (*e.g.*, $r = 30$): $P_{m\times m} \approx U_{m\times r} \cdot V^T_{m\times r}$ While proximity embedding was originally designed for centralized social network analysis [76], once the decomposition $P \approx UV^T$ becomes available, we can immediately use U and V to enable efficient, distributed computation of social proximity. Specifically, each node i is associated with a pair of vectors $U[i, *]$ and $V[i, *]$, which we term as i's *social coordinates* since they represent a user's position in the social network. The proximity from node i to node j can then be approximated as the dot product of their social coordinates $U[i, *]$ and $V[j, *]$, which can be efficiently computed in $O(r)$ time. Similarly, the proximity from node j to node i is simply the dot product of $U[j, *]$ and $V[i, *]$.

In addition, dot product can be used to compute many other proximity measures. For example, given a global list of attributes, each user is assigned a vector with 0s or 1s, where 0 means that the user does not have the attribute and 1 otherwise. The dot product of two vectors essentially captures the number of matches between two users and can also be used to predict new friendship. When the list gets too long, a bloom filter can be used to summarize

the list in a compact fashion. One can then estimate the similarity between two sets based on the dot product of the corresponding two bloom filters. As another example, consider two feature vectors $\mathbf{u}$ and $\mathbf{v}$. If we normalize them to have unit length, the dot product $\mathbf{u} \circ \mathbf{v}$ gives the cosine similarity between the two feature vectors.

A main component of our solution for secure proximity computation is the first secure dot product protocol that is both privacy-preserving and verifiable. Since secure dot product is a fundamental primitive in privacy-preserving data mining and secure multi-party computation, our technique can have many applications beyond mobile social networks.

Problem formulation. The social coordinates for individual users can be precomputed by a trusted central server in the social network that users belong to. For example, one can imagine that existing online social networking site, such as Facebook and MySpace, to provide such a service. Our goal then is to determine whether the dot product of two users' social coordinates exceeds a given threshold. Only when the dot product is large enough, will the two users start further communication. We want such computation to be efficient, privacy-preserving, and verifiable: (i) no user can forge social coordinates or the result of dot product without getting caught, and (ii) if the dot product between two users is below the threshold, they learn only this fact and nothing more.

2.3 Attack Models

Adversaries are curious about other users' personal information and location information and will try their best to extract information from every message. They may also lie or even collude in order to get more information. Below we identify a range of potential attacks against coordinate-based social proximity computation and quantify their effectiveness using analysis of real traces.

Compromising location privacy. A number of attacks can be launched to breach a user's location privacy based on her social coordinate or her social proximity to a known social coordinate. We identify four such attacks below.

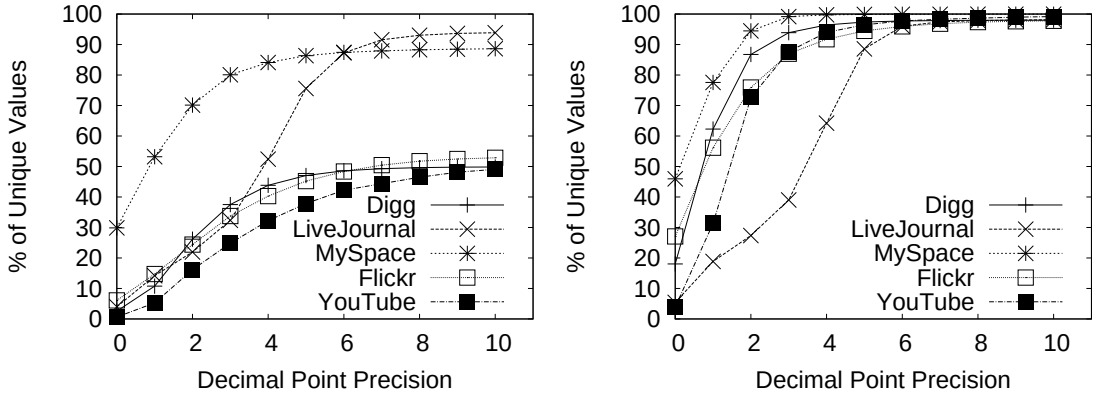
• *Fingerprinting users based on social coordinates.* A naive way to support proximity computation is to let every user publish her social coordinate for computing dot product. However, if the social coordinates of a user are relatively unique, it becomes easy to identify a user based on them. To assess the potency of such an attack, we use five popular social networks: Digg, Flickr, LiveJournal, MySpace and YouTube, as shown in Table 2.1. Figure 2.1(a) plots the percentage of unique social coordinates in a given social network as a function of the number of digits used to represent one dimension in each coordinate. The curve is based on the first snapshot but the results from the second snapshot are similar. It is evident from the figure that with just 4 decimal-point precision, which is required for accurately computing proximity metric between any user pair, 35%-80% of the users become uniquely identi-

fiable by their social coordinates. Even with 1 decimal point precision, there are still 5%-50% unique social coordinates. Further inspection suggests that most non-unique social coordinates belong to socially inactive users with few friends. For example, all users with zero friends have the same all-zero social coordinate. To illustrate this effect, Figure 2.1(b) plots the percentage of unique social coordinates for users with at least five friends. We see a dramatic increase in the fraction of unique social coordinates. Therefore, social coordinate based fingerprinting is more potent against socially active users.

Network	Date	# Connected nodes	Links
Digg	9/15/2008	535,071	4,432,726
	10/25/2008	567,771	4,813,668
Flickr	3/01/2007	1,932,735	26,702,209
	4/15/2007	2,172,692	30,393,940
LiveJournal	11/13/2008	1,769,493	61,488,262
	12/05/2008	1,769,543	61,921,736
MySpace	12/11/2008	2,128,945	89,138,628
	1/11/2009	2,137,773	90,629,452
YouTube	4/30/2007	2,012,280	9,762,825
	6/15/2007	2,532,050	13,017,064

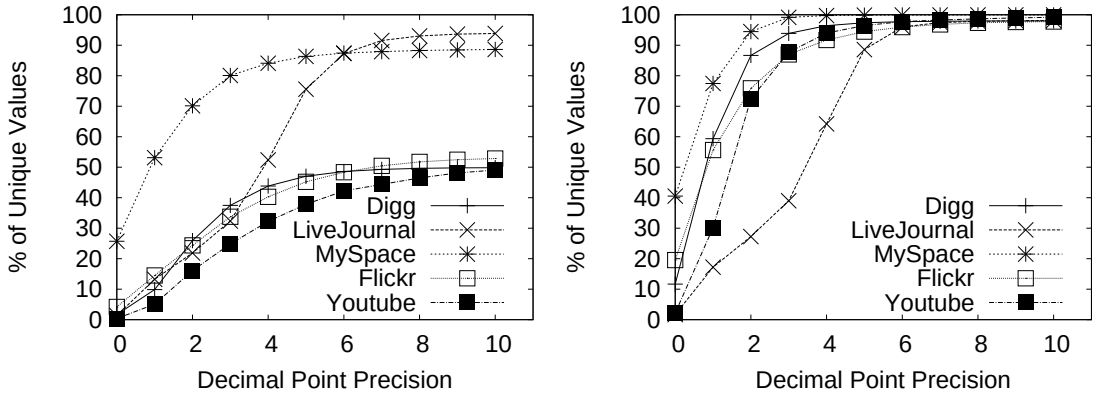
Table 2.1: Dataset summary.

• *Fingerprinting users based on social proximity.* Even if the social coordinates are encrypted, a smart adversary can try to infer the social coordinates based on the proximity metric. This is demonstrated by the following analysis. Figure 2.2 shows the uniqueness of the dot product value between a users' social coordinate and a given random coordinate that an attacker would use. As we can see, with 1 decimal point precision, 5-55% of all users are uniquely identifiable; as the precision increases to 4 decimal points, around 30-80% of the users are uniquely identifiable. Moreover, the fraction of unique



(a) all users (b) users with ≥ 5 friends

Figure 2.1: Percentage of unique social coordinates as a function of precision.



(a) all users (b) users with ≥ 5 friends

Figure 2.2: Percentage of unique dot products as a function of precision.

social proximity is much higher among users with at least five friends. So social proximity based fingerprinting is a more serious attack against socially active users.

• *Inferring social coordinates from proximity measure.* Apart from acting as an identifier, knowing exact proximity to a given user may even allow an adversary to reconstruct a user's social coordinates. Specifically, since the

proximity between two users is a dot product between their coordinates, each proximity value gives one linear constraint on the social coordinates of the other party. An adversary just needs d linearly independent constraints to reconstruct a d-dimension social coordinate. This can be achieved by having an adversary generate d (fake) linearly independent coordinates and then compute the proximity with an intended victim, using each of the d different coordinates or having d colluding adversaries.

• *Binary search on social proximity.* The above attacks suggest that we should protect privacy of both social coordinates and proximity measure. In order to facilitate the decision of whether to make friends with a user, ideally we only want to reveal 1 bit of information, *i.e.*, whether the dot product is above or below a threshold (instead of its exact value). However, even this 1 bit of information could be potentially exploited by a patient adversary to launch a binary search attack that adaptively adjusts the threshold to quickly narrow down the value range for the social proximity between the victim and a social coordinate chosen by the adversary.

Tracking users. We further observe that social coordinates are relatively stable over time. Figure 2.3 plots the cumulative distribution function of the cosine similarity (which measures the angle between two vectors) between a user's social coordinates across the two snapshots indicated in Table 2.1 for all the social networks. As shown in Figure 2.3, for most networks, a large fraction of users' coordinates have high cosine similarity across the two snapshots. 50-60% people in Digg and Flickr have social coordinates' cosine

similarity as high as 0.9, and about 90% of users in MySpace have cosine similarity over 0.9. Such high stability in social coordinates indicates that an adversary could potentially link social coordinate announcements at different times, locations, and networks to the same user and enable continuous user tracking.

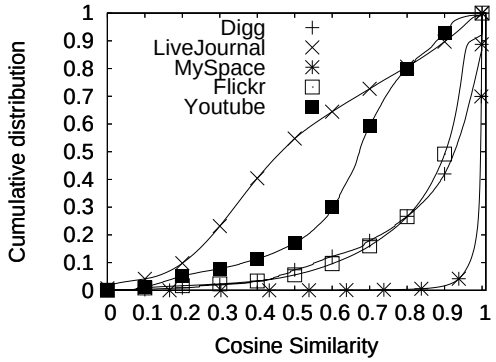


Figure 2.3: CDF of cosine similarity of social coordinates in two snapshots.

Other attacks. Other attacks include forgery and DoS. Since users rely on the social coordinates to make friendship decisions, it would be very damaging if an adversary is able to lie about her social coordinate to trick others in to believing that the adversary is socially close. Also existing private dot product computation protocols are not verifiable (*e.g.*, [38, 81]), making it possible to lie about the computation result. Finally, an adversary may send many fake social coordinates for proximity computation and cause overloading at the victim.

Chapter 3

Secure Friend Discovery Protocols

3.1 Design Goals

Based on the above attacks, our design goals are as follows:

1. **Preserving the privacy of social coordinates.** Due to the uniqueness of social coordinates, we should encrypt social coordinates. Since each dot product reveals one linear constraint about the social coordinates, d independent constraints would reveal the whole coordinate. Therefore we should limit the number of linear constraints revealed.

2. **Preserving the privacy of social proximity.** Due to uniqueness of proximity values and the possibility of reconstructing social coordinates based on proximity values, ideally a scheme should just reveal whether the proximity value is below or above a threshold used for making friendship decisions, while protecting against binary search attacks.

3. **Preventing user tracking.** Even when a user's social coordinate does not change, her social coordinate announcement should look different to prevent tracking.

4. **Providing authentication and verification.** Users should not be able to forge their social coordinates. Any user should be able to authenticate another user's identity and verify if the social coordinate used for computation is valid.

5. **Efficient filtering.** Since a user is interested in quickly finding potential friends and the number of potential friends is usually much smaller than the total number of users in the network, we should efficiently filter out social coordinates with low proximity values.

3.2 Overview

To achieve the above design goals, we develop a novel secure proximity computation protocol, which consists of three major components: (i) authentication without long-term linkability (Section 3.3), (ii) efficient and privacy-preserving proximity pre-filtering (Section 3.4), and (iii) private and verifiable proximity computation (Section 3.5).

In (i), we leverage the idea of virtual ID to remove long-term linkability and prevent user tracking, and we use digital signatures for authentication. The proximity prefiltering in (ii) allows users to quickly filter out users that are unlikely to become friends. However it does not prevent an adversary from forging social coordinates or the final proximity result. So we further develop a technique for private and verifiable proximity computation based on homomorphic cryptography in (iii) to check the validity of social coordinates

and proximity result.

The step (iii) is only invoked when proximity pre-filtering determines that the proximity exceeds the threshold because (i) homomorphic cryptography is computationally expensive and should be called upon only when necessary, for verification, and (ii) this step reveals the exact proximity value. It is unacceptable to reveal the exact proximity value to an arbitrary user, which can easily compromise location privacy (see Section 2.3). However, we believe that it is acceptable to reveal the exact proximity value to those users that are sufficiently close socially and thus likely to become friends. A lying adversary can get caught by this step and reported.

3.3 Authentication Without Long-Term Linkability

When two strangers encounter each other, proximity computation needs to be performed on their social coordinates though they may not trust each other. To support authentication in such an untrusted environment, we require a trusted server, which can be the same server that computes social coordinates. Note that the server is only used for bootstrapping trust. *It does not need to be contacted in a mobile social network.*

To protect Alice's identity and prevent long-term linkability, the trusted server assigns Alice a separate *virtual ID*, which is the only ID sent during communication. The virtual ID is essentially a temporary private/public key pair, which is valid for only a specified time period. Alice needs to get her new virtual ID by contacting the trusted server when it is reachable. Every

time Alice contacts the server, the server can use standard digital signature techniques to authenticate Alice (as in Internet server) before issuing her new virtual ID.

After authenticating Alice, the server sends Alice her new virtual ID, expiration time, encrypted social coordinates, and the server's digital signature. Others can use the server's signature and Alice's own signature to authenticate Alice and her encrypted social coordinates. The encrypted social coordinates are used for computation in a verifiable secure dot product protocol introduced in Section 3.5. In this way, we cannot link messages sent before and after changing virtual IDs, which prevents long-term linkability. To reduce the frequency of contacting the server, the server may issue Alice multiple virtual IDs at a time, each with a different valid period.

3.4 Proximity Pre-filtering

The goal of proximity pre-filtering is to quickly identify potential friends from all the users in a mobile social network. Our requirement is to efficiently compute whether the proximity between two social coordinates is below or above a threshold without revealing the actual social coordinates or the exact proximity. The threshold depends on a user's interest in making new friends. In this step, we do not verify the correctness of the coordinates or proximity values, which will be handled by the private and verifiable proximity computation in Section 3.5.

There are several existing protocols that compute dot product while

preserving the privacy of the vectors (*e.g.*, [38, 81]), however they reveal the final value of the dot product to one of the parties and do not satisfy our requirements (*i.e.*, preserve the privacy of the proximity value and only reveal whether it is above or below the threshold). We first review one of the existing secure dot product proposed [38], which we use as the basis for our protocol. Then we present our modifications to achieve our design goal.

Existing secure dot product protocol [38]. Suppose Alice has a d-dimensional vector $\mathbf{v}$ and Bob has a d-dimensional vector $\mathbf{u}$. s is a security parameter that controls the amount of random information we use to hide the social coordinate.

1. Alice initiates the dot product computation by asking Bob to start the computation. Bob constructs a $s \times (d+1)$ matrix $\mathbf{X}$. Its i-th row $\mathbf{x_i}$ is defined as

$$\mathbf{x_i} = \begin{cases} \langle u_1, u_2, \ldots, u_d, 1 \rangle, & i = r, \\ \mathbf{k_i}, & \forall i \neq r, \end{cases} \tag{3.4.1}$$

   where r is a randomly chosen row, which contains the social coordinate, and the other row vectors $\mathbf{k_i}$ $(i \neq r)$ are all randomly generated.

   Bob also creates an $s \times s$ random matrix $\mathbf{Q}$, a random $(d+1)$-dimensional vector $\mathbf{f}$, and three random numbers r_1, r_2, r_3. He computes the following five terms: (i) $b = \sum_{i=1}^{s} Q_{ir}$, (ii) $\mathbf{c} = \sum_{i=1, i\neq r}^{s} (\mathbf{x_i} \cdot \sum_{j=1}^{s} Q_{ji})$, (iii) $\mathbf{Q} \cdot \mathbf{X}$, (iv) $\mathbf{c}' = \mathbf{c} + r_1 \cdot r_2 \cdot \mathbf{f}$, and (v) $\mathbf{g} = r_1 \cdot r_3 \cdot \mathbf{f}$. Bob then sends $\mathbf{Q} \cdot \mathbf{X}, \mathbf{c}', \mathbf{g}$ to Alice.

2. Alice chooses a random number α and creates a vector $\mathbf{v}' = \langle v_1, v_2, ...v_d, \alpha \rangle$. Alice then computes the following four terms: (i) $\mathbf{y} = \mathbf{Q} \cdot \mathbf{X} \cdot \mathbf{v}'$, (ii) $z = \sum_{i=1}^{s} y_i$, (iii) $a = z - \mathbf{c}' \circ \mathbf{v}'$, and (iv) $h = \mathbf{g} \circ \mathbf{v}'$. Here $\mathbf{v}_1 \circ \mathbf{v}_2$ represents the dot product (*i.e.*, inner product) of two given vectors $\mathbf{v}_1$ and $\mathbf{v}_2$. Alice then sends a and h to Bob.

3. Bob computes $\beta = \frac{a+h\cdot(r2/r3)}{b}$ and sends it to Alice.

4. Alice computes $\beta - \alpha$, which is the desired dot product as shown in [38] (*i.e.*, $\beta - \alpha = \mathbf{v} \circ \mathbf{u}$).

Proximity pre-filtering. Since in the last step of the above protocol the dot product is given by $\beta - \alpha$, a simple way to hide the true dot product is to let Alice choose a threshold T and send $T + \alpha$ to Bob. If β is larger than $T + \alpha$, Bob replies YES; otherwise he replies NO. The problem with this approach is that Bob can guess a reasonable T in this system, and with $T + \alpha$ and β he can estimate $\beta - \alpha$, which is the dot product.

To address this issue, we propose a proximity pre-filtering protocol by making the following modifications to the original secure dot product protocol. At the beginning of the protocol, instead of using her real vector, Alice uses a scaled version of her vector $\rho \cdot \mathbf{v}$, where ρ is a random number chosen by Alice. At the end of the protocol, along with a and h, Alice also picks another random number ρ' such that $0 \leq \rho' < \rho$, computes $\gamma = \alpha + \rho \cdot T + \rho'$ and sends γ to Bob. Bob compares γ with β, and replies YES only when $\beta > \gamma$, and otherwise replies No. All the other operations remain the same as [38].

Correctness. Since Alice scales her vector by ρ, we have $\beta - \alpha = \rho \cdot p$, where p is the dot product. $\beta - \gamma = \beta - \alpha - \rho \cdot T - \rho' = \rho \cdot (p - T) - \rho'$. Given p and T are both integers, we have that $\beta > \gamma$ implies $p > T$ and $\beta < \gamma$ implies $p \leq T$.

Setting the threshold. Allowing users to set an arbitrary threshold T is undesirable since users do not have the knowledge and it complicates liar detection. We assume a system suggests minimum threshold for all users. Users can set a higher threshold to be more selective but are prohibited from setting lower thresholds than the suggested value. Therefore whenever the outcome of proximity prefiltering is YES the proximity value must be larger than this suggested minimum threshold, which can be used as groundtruth for liar detection.

3.5 Private and Verifiable Proximity Computation

Proximity pre-filtering alone is insufficient because adversaries can falsify their social coordinates or the final proximity result. In this section, we develop a solution to address these issues. It achieves both privacy and verifiability, which are two conflicting goals in secure multi-party computation. We first introduce an existing homomorphic protocol for computing a dot product. It preserves privacy but does not provide authentication or verification. Then we describe our approach to provide both privacy and verification.

Homomorphic encryption. Homomorphic encryption is an effective solu-

Public key: (g, n), Private key: σ
Encryption:
Plaintext $m < n$
Select random $r < n$
Ciphertext $c = E(m, r) = g^{m+nr} mod\ n^2$
Decryption:
Ciphertext $c < n^2$
Plaintext $m = D(c) = \frac{L(c^\sigma\ mod\ n^2)}{L(g^\sigma\ mod\ n^2)} mod\ n$, where $L(u) = \frac{u-1}{n}$

Figure 3.1: The fast variant of Paillier's Cryptosystem

tion to privacy-preserving computation. It allows certain algebraic operations on the plaintext to be performed using (possibly different) algebraic operations directly on the ciphertext. There are several homomorphic cryptosystems. We use the fast variant of Paillier's cryptosystem [63], shown in Figure 3.1. Its semantic security relies on the premise that the Decisional Partial Discrete Logarithm Problem [63] is hard. This cryptosystem has the following two useful properties:

1. *Additive Homomorphic Property*: Multiplication of the ciphertexts results in addition of the plaintexts:

$$\begin{aligned} \mathsf{E}(m_1, r_1)\,\mathsf{E}(m_2, r_2) \mod n^2 &= \mathsf{E}(m_1 + m_2, r_1 + r_2) \\ \mathsf{E}(m_1, r_1)^{m_2} \mod n^2 &= \mathsf{E}(m_1 \cdot m_2, r_1 \cdot m_2) \end{aligned}$$

2. *Self-Blinding Property*: Any ciphertext can be changed to another without affecting the plaintext:

$$\mathsf{D}(\mathsf{E}(m, r_1)) = \mathsf{D}(\mathsf{E}(m, r_1 + r_2))$$

For our protocol, we expect the trusted server to generate the required keys and send them to the user in addition to the virtual ID information specified in Section 3.3.

Homomorphic dot product protocol. A secure dot product protocol has been proposed based on homomorphic encryption [29]. It achieves the design goal that Alice gets the dot product, but neither Alice nor Bob learns any other useful information about other party's vector, under certain security assumptions. To summarize, let H_A^+ and H_B^+ denote Alice and Bob's homomorphic public keys, respectively. Suppose Alice's vector is $\mathbf{v} = \langle v_1, v_2, \ldots, v_d \rangle$ and Bob's vector is $\mathbf{u} = \langle u_1, u_2, \ldots, u_d \rangle$. The protocol works as follows:

1. For each dimension of vector $\mathbf{v}$, Alice generates a random number r_i and sends $\mathsf{E}_{H_A^+}(v_i, r_i)$ to Bob.
2. Bob computes $w = \prod_{i=1}^{d} \mathsf{E}_{H_A^+}(v_i, r_i)^{u_i}$ (we will refer to this operation as multiplication throughout our paper), generates a random number r', and sends $w' = w \cdot \mathsf{E}_{H_A^+}(0, r')$ back to Alice.
3. Alice then decrypts w' to get the dot product.

This protocol is provably privacy-preserving. However, it assumes an *honest-but-curious* adversary model and provides neither authentication nor verification. Malicious participants can lie about their encrypted vectors and the resulting dot product of the two vectors, since Bob knows Alice's public key and can encrypt an arbitrary value to send back to Alice instead of the real dot product. In the rest of the paper we refer to this protocol as *Protocol 0*.

Designing a verifiable secure dot product protocol. Authentication and verification are essential to guard against malicious users who falsify the

social coordinates and proximity metric. Section 3.3 describes authentication, and below we discuss how to make the protocol verifiable.

Note that for both parties to obtain the dot product, both Alice and Bob run two separate instances of protocol in parallel. Then, a naïve verification approach for Bob may be to first decrypt the result sent by Alice using his private key and encrypt it using Alice's public key and compare if it with w that he computed before for consistency. However, since encryption in Paillier's cryptosystem involves a random number, the same plaintext does not generate the same ciphertext, causing this approach to fail.

Suppose Alice picks $\mathbf{r_1} = \langle r_{11}, r_{12}, \ldots, r_{1d} \rangle$ as her random numbers and Bob picks $\mathbf{r_2} = \langle r_{21}, r_{22}, \ldots, r_{2d} \rangle$ as his random numbers. Let r'_1 and r'_2 be the numbers used for self-blinding by Alice and Bob, respectively. Alice's encrypted vector is $\mathsf{E}_{H_A^+}(\mathbf{v}, \mathbf{r_1}) = \langle \mathsf{E}_{H_A^+}(v_1, r_{11}), \mathsf{E}_{H_A^+}(v_2, r_{12}), \ldots, \mathsf{E}_{H_A^+}(v_d, r_{1d}) \rangle$, and Bob's encrypted vector $\mathsf{E}_{H_B^+}(\mathbf{u}, \mathbf{r_2})$ is similar.

In order to solve the verification problem, we observe that when the fast variant of Paillier's cryptosystem is used, the value computed by Alice (before self-blinding) is $\mathsf{E}_{H_B^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_2} \circ \mathbf{v})$. This is because

$$\prod_{i=1}^{d} \mathsf{E}_{H_B^+}(u_i, r_{2i})^{v_i} = \prod_{i=1}^{d} \mathsf{E}_{H_B^+}(u_i \cdot v_i, r_{2i} \cdot v_i) = \mathsf{E}_{H_B^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_2} \circ \mathbf{v})$$

Assume Alice decrypts the result received from Bob and gets $result1$, without knowing $\mathbf{r_2} \circ \mathbf{v}$, Alice cannot generate $\mathsf{E}_{H_B^+}(result1, \mathbf{r_2} \circ \mathbf{v})$ to test the consistency with $\mathsf{E}_{H_B^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_2} \circ \mathbf{v})$ computed by herself.

To overcome such difficulty, below we design two protocols that support verification and at the same time let users use different random vectors for encryption.

Verifiable secure dot product protocol 1: Here, we apply Protocol 0 to compute $\mathbf{r_2} \circ \mathbf{v}$ in addition to $\mathbf{v} \circ \mathbf{u}$ so that the dot product value obtained can be verified.

1. As in Protocol 0, Alice and Bob start by exchanging their encrypted vectors $\mathsf{E}_{H_A^+}(\mathbf{v}, \mathbf{r_1})$ and $\mathsf{E}_{H_B^+}(\mathbf{u}, \mathbf{r_2})$.
2. Alice computes $\mathsf{E}_{H_B^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_2} \circ \mathbf{v})$ and $\mathsf{E}_{H_B^+}(\mathbf{r_1} \circ \mathbf{u}, \mathbf{r_1} \circ \mathbf{r_2})$ and send them to Bob after self-blinding. Bob computes $\mathsf{E}_{H_A^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_1} \circ \mathbf{u})$ and $E_{H_A^+}(\mathbf{r_2} \circ \mathbf{v}, \mathbf{r_1} \circ \mathbf{r_2})$ and also send them after self-blinding.
3. Alice decrypts and gets two numbers $result1$ and $result2$, which are supposed to be $\mathbf{v} \circ \mathbf{u}$ and $\mathbf{r_2} \circ \mathbf{v}$. Alice computes $\mathsf{E}_{H_B^+}(result1, result2)$ and compares with $\mathsf{E}_{H_B^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_2} \circ \mathbf{v})$. If they are consistent, the dot product $result1$ is correct. Bob decrypts and verifies in the same way.

To ensure that Alice can properly decrypt $\mathbf{v} \circ \mathbf{u}$ and $\mathbf{r_2} \circ \mathbf{v}$ in step 3 of the protocol, we require that $\mathbf{v} \circ \mathbf{u} < n_A$ and $\mathbf{r_2} \circ \mathbf{v} < n_A$, where $(g_A, n_A) = H_A^+$ is Alice's homomorphic public key given in Figure 3.1. This can be achieved by limiting the number of bits of each element in $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{r_2}$. In our current implementation, n_A has 1024 bits, each element in $\mathbf{u}$ and $\mathbf{v}$ is a 32-bit integer, and the number of dimensions d is a 16-bit integer. As a result, each element in $\mathbf{r_2}$ can have as many as $1024 - 16 - 32 = 976$ bits without causing $\mathbf{r_2} \circ \mathbf{v}$ to

exceed n_A. Such a large number of bits suffice to defend attacks that enumerate all possible ciphertexts for certain plaintext in a brute-force fashion.

Compared to the original Protocol 0, our new protocol has slightly more communication overhead and more than twice computation overhead since it computes two dot products using the Protocol 0 and has a verification phase, which is essentially an encryption. Also note that the second dot product incurs more computation overhead because the elements of the random vectors can be much larger than elements of $\mathbf{v}$ and $\mathbf{u}$. Therefore, we call the multiplication operation with random vector as *BigMul*. Next we develop a light-weight protocol with much less computation overhead, but slightly more communication overhead.

Verifiable secure dot product protocol 2: The idea is to only compute $\mathbf{v} \circ \mathbf{u}$ using Protocol 0 and compute $\mathbf{r_2} \circ \mathbf{v}$ using the secure dot product protocol [38], which is much cheaper than the homomorphic protocol.

1. Alice and Bob exchange their encrypted vectors $\mathsf{E}_{H_A^+}(\mathbf{v}, \mathbf{r_1})$ and $\mathsf{E}_{H_B^+}(\mathbf{u}, \mathbf{r_2})$ as before.
2. Alice computes $\mathsf{E}_{H_B^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_2} \circ \mathbf{v})$ and sends it to Bob after self-blinding. Bob computes $\mathsf{E}_{H_A^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_1} \circ \mathbf{u})$ and does the same thing.
3. Alice uses the secure dot product protocol [38] to compute $\mathbf{v} \circ \mathbf{r_2}$ with Bob. We denote the result she gets from this step as $result2$. Similarly, Bob uses secure dot product protocol to compute $\mathbf{u} \circ \mathbf{r_1}$.

4. Alice decrypts the received message from step 2 and get $result1$. With $result1$ and $result2$, verification is the same as Protocol 1. The same holds for Bob.

Compared to Protocol 0, the additional computation cost is close to just one more encryption for verification, since the additional secure dot product computation is much cheaper than the encryption. In terms of communication, the secure dot product protocol requires at least four messages, which slightly increases communication cost. Since the computation cost dominates, the small increase in communication is worthwhile.

Homomorphic encryption with negative numbers: Negative numbers in the social coordinates do not affect the correctness of the protocols since the homomorphic properties still hold. A negative dot product can cause ambiguity on the results because the decrypted result x may imply that the original result may be x, $x - n$, or $x - 2 \cdot n$, etc. To handle this issue, we leverage the fact that n is a 1024-bit number, which is huge, whereas all proximity values comparatively small. Therefore, whenever the proximity is too large (*i.e.*, $> n/2$), this indicates that the real value is negative and the actual result is $result - n$.

3.6 Security Analysis

3.6.1 Proximity Pre-filtering

Here we analyze the information flow between Alice and Bob. The first several steps of our protocol are the same as the secure dot product protocol, except that we scale the coordinate. As shown in [38], from Bob to Alice, $\mathbf{Q} \cdot \mathbf{X}$, $\mathbf{c}'$ and $\mathbf{g}$ do not leak any information. From Alice to Bob, h reveals one equation about Alice's coordinate $\mathbf{v}$. This is possible because we have two relationships $h = \mathbf{g} \circ \mathbf{v}'$ and $\beta = p + \alpha$; canceling out α from both relationships give us one constraint on $\mathbf{v}$. Now we consider the additional information flow introduced in our protocol. The final result from Bob to Alice contains 1 bit information, which is the outcome of the protocol and required. From Alice to Bob, γ reveals $\rho \cdot (p - T) - \rho'$. So essentially we hide $p - T$ using two random numbers ρ and ρ'. [43] used the same hiding technique in Yao's millionaire's protocol, which is shown to be secure when the range of ρ is big enough and one can increase the difficulty of guessing $p - T$ by randomly using many possible distributions for choosing random numbers. It will reveal $p = T$ only when $\rho \cdot (p - T) - \rho' = 0$, which occurs with a very low probability.

Next we examine if binary search is still possible. There are two ways to perform binary search: (i) an adversary computes proximity measure with the victim multiple times, each time using a different threshold, or (ii) an adversary colludes with others and fakes the same social coordinate and compute proximity measure between the fake coordinate with the victim's coordinate multiple times, each time using a different threshold. The first attack can be

easily prevented by permitting only one proximity computation between any two users. To defend the second attack, we note that in binary search about half of the trials will have YES result. Whenever the result is YES, we run our verifiable secure dot product protocol, which will detect the liar. So binary search is still possible, but each trial will be caught with 50% probability and the success probability of binary search exponentially decreases with the number of trials involved in binary search.

The pre-filtering protocol does not have authentication or verification on the vectors used by participants. It is easy for either side to manipulate the final result. Alice can manipulate γ and make the result NO even when the real result is YES. This means that Alice decides not to make friend with Bob despite the high proximity measure, which should be permitted as Alice should have freedom to choose friends even when the proximity measure exceeds Bob's threshold. Similarly, Bob is allowed to reply NO when the actual result is YES. But anyone that makes the result YES while it is actually NO is considered cheating and can be detected by our verifiable secure dot product protocol.

3.6.2 Private and Verifiable Proximity Computation

Theorem 1 (Privacy of Coordinates in Protocol 1)**.** *Assuming the fast variant of Paillier's cryptosystem is semantically secure, Alice and Bob only get the dot product and no more useful information about each other's coordinate.*

Proof. Assuming the cryptosystem is semantically secure, Protocol 0 reveals no useful information other than the result [29]. In Protocol 1, each participant

gets two dot product by running two instances of Protocol 0. Thus each of them gets no more useful information than two dot products. The extra dot product used for verification, *i.e.*, $\mathbf{r_2} \circ \mathbf{v}$, does not reveal more information about $\mathbf{u}$ because $\mathbf{r_2}$ is hard to guess. In fact, guessing $\mathbf{r_2}$ based on $\mathbf{r_2} \circ \mathbf{v}$ is just as hard as guessing $\mathbf{u}$ based on $\mathbf{v} \circ \mathbf{u}$. □

Theorem 2 (Privacy of Coordinates in Protocol 2)**.** *Assuming the fast variant of Paillier's cryptosystem is semantically secure, Alice and Bob get the dot product and one more linear constraint about the other party's coordinate.*

Proof. The only difference between Protocol 1 and Protocol 2 is that in Protocol 2 $result2$ is computed using secure dot product protocol [38], which reveals one more constraint about the other user's coordinate. □

Theorem 3 (Verification Guarantee)**.** *If Alice's verification is successful in Protocol 1 or Protocol 2, Alice gets the real dot product.*

Proof. The only case when verification is successful but Alice gets a wrong number is Bob found two numbers $result1 < n$ and $result2 < n$ such that $result1 \neq \mathbf{v} \circ \mathbf{u}$, $\mathsf{E}_{H_B^+}(result1, result2) = \mathsf{E}_{H_B^+}(\mathbf{v} \circ \mathbf{u}, \mathbf{r_2} \circ \mathbf{v})$. This is impossible because decryption result is unique in Paillier's cryptosystem. □

Chapter 4

Friend Discovery Protocol Evaluation

We implement our protocols on HP IPAQ 910, which has Marvell PXA270 416 MHz Processor, 128 MB RAM, Windows Mobile 6.1 Professional operating system, 802.11 b/g WiFi card, and .NET Compact Framework. Since the cryptosystem involves very large numbers, which is not supported by the Compact Framework, we use a public BigInteger Library for C# [16]. The communication uses 802.11b ad hoc mode. We also implemented all computations in all the protocols on Motorola Droid. It has 550 MHz Arm Cortex A8 processor, 230 MB RAM, and uses Java and the Android development toolkit. We use a publicly available implementation [64] of Paillier's Cryptosystem in Java as a basis for our Fast Variant implementation. Since the Android Platform has not yet supported wireless ad-hoc mode, we quantify the communication cost using 802.11b infrastructure mode. For comparison, we also evaluate our protocols on a laptop PC with Windows Vista system, P8600 processor, and 2GB memory. Our implementation uses Java. We perform power measurement on the Droid using PowerTutor [69]. All the implementations use 30-dimension social coordinates.

Device	Mean	Max	Min	Std	Median
PC	0.44	2.15	0.29	0.30	0.32
IPAQ	172	633	44	78	154
Droid	44	112	22	23	23

Table 4.1: Execution time of proximity prefiltering (ms)

4.1 Proximity Pre-filtering

The efficiency of pre-filtering depends on the security parameter s, which controls the amount of random information we add. We use $s = 2$ in our evaluation as in [38].

This protocol altogether involves 4 messages: (i) a control message to initiate proximity pre-filtering, (ii) a reply containing a matrix and two vectors, which has altogether $(s + 2) \cdot (d + 1)$ numbers, (iii) a response containing 3 numbers, and (iv) a final answer containing 1-bit: either YES or NO to the question whether the proximity exceeds the threshold. The message size depends on the selected random numbers.

Table 4.1 shows the mean, maximum, minimum and average execution time for prefiltering over 100 runs, for PC, IPAQ and Droid. The average time is 0.17 second on the IPAQ and 0.044 second on the Droid. Both times are fast enough for practical use.

4.2 Private and Verifiable Proximity Computation

We implement both versions of our verifiable secure dot product protocols. Our implementation is based on the fast variant of the Paillier's cryp-

PC					
Operation	Mean	Max	Min	Std	Median
Authentication	0.48	1.31	0.37	0.12	0.46
Encryption	77.79	96.07	76.77	2.23	77.20
Decryption	14.11	17	13	0.5667	14
Multiplication	81.92	124	78	5.25	81
BigMul	2253.8	2581	2223	52.02	2238
Self-blinding	83	120	81	4.73	82
Verification	82.43	93	80	2.27	82
IPAQ					
Operation	Mean	Max	Min	Std	Median
Authentication	27.02	61	24	7.7485	25
Decryption	2193.2	2620	2176	48.3319	2178
Multiplication	4055.2	6033	2312	1134.7	4146
BigMul	382026	390237	375463	7523.6	380378
Self-blinding	12615	13353	12280	196.74	12606
Verification	12807	14270	12452	240.10	12776
Droid					
Operation	Mean	Max	Min	Std	Median
Authentication	2.54	54.4	2.23	3.01	2.32
Decryption	27.01	63	24	8.6416	24
Multiplication	160.41	554	150	44.5594	153
BigMul	3780.7	4250	3733	71.48	3760.5
Self-blinding	144.69	596	136	47.6366	137
Verification	144.18	192	136	14.1581	138

Table 4.2: Breakdown of computation time (ms)

tosystem, where n has 1024 bits and σ has 160 bits according to [63]. The random numbers used have 900 bits, big enough to prevent brute force chosen plaintext attacks.

Table 4.2 shows the breakdown of the computation time on PC, IPAQ, and Droid. Protocol 1 involves one authentication, one multiplication, one BigMul, two self-blindings, two decryptions, and one verification. Protocol 2 includes one authentication, one multiplication, one self-blinding, one decryption, one secure dot product protocol [38], and one verification. We do not

Protocol	Power Consumption (mJ)
Announcement - precompute	0.4773
Announcement - identify	2.907
Proximity Prefiltering	43.388
Protocol 1	2277.9
Protocol 2	286.3608

Table 4.3: CPU power consumption on Droid

show the performance of secure dot product protocol because it has almost the same performance as the proximity pre-filtering shown in Table 4.1. As we would expect, the PC is faster than Droid, which is faster than IPAQ. The average computation time for protocol 1 is 2.61 seconds on PC, 4.4 seconds on Droid, and 6 minutes on IPAQ. The time for protocol 2 is 0.276 seconds on PC, 0.523 seconds on Droid, and 31.869 seconds on IPAQ. The difference between computation time on IPAQ and Droid comes from two factors: (i) the IPAQ is older and has a slower processor, and (ii) more importantly, the built-in Java BigInteger implementation is much more efficient than C# implementation [16] for the IPAQ. Microsoft is planning to introduce built-in BigInteger class in their new .NET framework and we expect the running time on IPAQ can reduce significantly with a more efficient library. The server pre-computes vector encryption on behalf of a mobile host. The time to perform encryption for a single dimension on PC is 77.79 ms, as shown in Table 4.2.

Table 4.3 shows CPU power consumption for running protocol 1 and 2 on Droid. To put the number into perspective, a fully charged Droid has 18,648,000 mJ. So the computation involved in the two protocols consume 0.0122% and 0.0015% total power, respectively. These numbers indicate that

both protocols are practical in terms of power, as well.

In terms of operation, BigMul is the most expensive since it includes d 900-bit random numbers. Self-blinding cost also depends on the random number used (900 bits). In our evaluation, we do not use negative numbers. While negative numbers affect the performance of multiplication (since raising a negative power requires an inverse operation), their influence can be eliminated by letting the server pre-compute the inverse of each dimension in the encrypted vector and giving them to users. This increases the communication cost between users without leaking more information.

In terms of message sizes, for protocol 1, request message contains the homomorphic public key (n, g) and encrypted vector. n is a 1024-bit number, and g has at most 2048 bits. The encrypted vector has d encrypted numbers, each with at most 2048 bits. Hence the total request message is at most 8064 bytes for 30-dimensional social coordinates. The reply message contains two encrypted numbers, each having at most 2048 bits. So the total size is within 512 bytes. Protocol 2 contains only one encrypted number in the reply but involves 4 additional messages as in proximity prefiltering (described in Section 4.1) except the last message contains the dot product value instead of YES or NO, and Alice does not send γ, and the numbers are much bigger because one participant is now using the random vector. The total message size in our implementation is around 8900 bytes for protocol 1 and around 16K bytes for protocol 2. The exact value may vary depending on the encryption result and the random numbers.

The largest message among all protocols is the protocol 1 request message (8K bytes), which takes 3.96 ms and 0.195 mJ to transmit on average over 200 runs on the Droid. Hence, the communication cost for the protocols is not significant compared to the computation costs.

Chapter 5

iDEAL: Incentivize Cellular Offloading via Auction

5.1 Problem Formulation

In this section, we formulate the problem of offloading cellular traffic as a *reverse auction*. The offloading is transparent to clients and does not affect cellular pricing (*i.e.*, users pay for the data usage regardless of whether it is carried by the cellular provider or third party resource owners).

Basic auction settings: Consider a cellular network A which is interested in purchasing and leveraging spare resources from third-party Wi-Fi hotspots to satisfy traffic demands from its customers. The third-party hotspot owners should be rewarded for opening up their services to A's customers. To facilitate such cooperation, provider A can set up an auction to let third-party hotspot owners submit bids to offer their network resources, *e.g.*, dollars per bit-rate for unit time (*e.g.*, 1 hour) that a third-party hotspot owner offers.

This problem is naturally formulated as a *reverse auction*. Since the demand changes over time, *e.g.*, due to diurnal variations [72], the auction takes place periodically or whenever demand changes. The auction frequency is chosen to balance the overhead and the accuracy of traffic demand estimation.

The cellular network is shared across a relatively large area typically called a cell site. A site is further sub-divided into three or more sectors. Cellular resource in different sectors is relatively independent so we only consider a single sector. Same solution can be applied independently to other sectors. The sector can be considered to be divided into m small regions based on locations of Wi-Fi hotspots and Wi-Fi range as shown in Figure 5.1. A Wi-Fi hotspot can satisfy traffic demands only in its region.

Naïve solution: A simple approach is to statically partition the cellular resource into different regions and determine the amount of Wi-Fi resource needed in each region based on the amount of user demand in the region. Then we conduct a local auction within a region to utilize the cellular resource and Wi-Fi resources dedicated to the region. We call it *static local auction*. While simple, this approach has several important limitations: 1) Due to limited Wi-Fi coverage, the number of hotspots in a region is limited, *i.e.*, the competition is limited. However, adequate competition is essential for an auction based approach to be effective. 2) This formulation treats different regions equally, however the service provider may view different regions differently because different regions may have different spectrum efficiencies due to different signal-to-interference-noise-ratio (SINR) from the base station. 3) The static allocation cannot effectively take into account the available Wi-Fi resources and their bids across different regions. For example, even when a region has higher traffic demand, we may or may not need to allocate more cellular resources to the region depending on (i) how many Wi-Fi hotspots

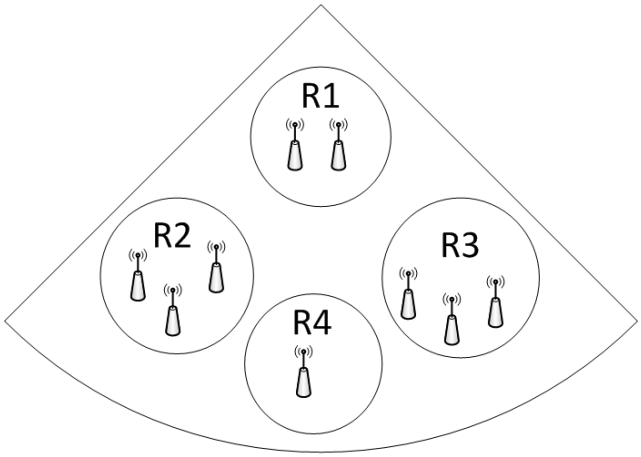


Figure 5.1: A sample cellular sector and its Wi-Fi regions

are in the region, (ii) what are their prices, and (iii) how the Wi-Fi hotspots and their prices compare with those in other regions. If there are more Wi-Fi hotspots in a region offering cheaper bids than in the other regions, we can allocate less cellular resources.

Design goals: We seek an auction scheme to (i) *account for different spatial coverage of resources*, which has not been considered in existing work, (ii) *cope with dynamic traffic demands*, (iii) *achieve high efficiency*, where the winners in the auction are the hotspot owners who really can provide the service at a cheaper price, thereby improving the overall system efficiency and social welfare, (iv) *promote truthful bidding* to prevent bidders from gaming the system, effectively discover price to ensure that the overall system is efficient, and avoid unnecessary system fluctuation due to gaming, as unwanted switching between Wi-Fi and 3G can negatively impact user experience [23], (v) *low cost*, which is natural but is challenging to achieve simultaneously with truthfulness, and

m	number of regions in a cellular sector
n	number of sellers in a cellular sector
d_i	traffic demand in region i
c_i	cellular capacity in region i
e_i	spectrum efficiency of cellular network in region i
z	total cellular spectrum usage: $z = \sum_i^m c_i/e_i$
x_j	total capacity bought from seller j
p_j	the unit price seller j asks for
λ_j	the Wi-Fi capacity offered by seller j
$F(z)$	cellular cost function
$f(j)$	the region that seller j belongs to

Table 5.1: Notations.

(vi) *guard against collusion.*

5.2 Our solution: iDEAL

In this section we introduce our solution: iDEAL. We start by designing the auction setting that fosters more competition and captures the service provider's regional preferences. Then we describe the two stages of iDEAL: (i) *allocation*, *i.e.*, determine how to allocate traffic among third-party resource owners and the cellular network itself to minimize cost given the bids, (ii) *pricing*, *i.e.*, decide how much should be paid to individual third-party resource owners in order to provide enough incentives for them to be truthful. Optimal allocation does not depend on pricing, but assumes all sellers are truthful. Pricing depends on the allocation and is designed so that staying truthful is the seller's optimal strategy. Table 5.1 summarizes the key notations.

5.2.1 iDEAL Auction Setting

Third-party Wi-Fi resources and bids: Suppose n third-party hotspot owners offer their resources to the cellular service provider by submitting their bids. Let $A_j = \{\lambda_j, p_j\}$ denote hotspot owner j's bid, which indicates hotspot owner j wants to sell λ_j amount of bandwidth at a price p_j per bit per second. The bids are *non-atomic* (*i.e.*, a hotspot owner is willing to sell a part of the capacity it offers). Function $f(j)$ returns the region where hotspot owner j sells its capacity (*e.g.*, $f(j) = i$ means hotspot owner j sells its capacity in region i). For simplicity, we assume that each hotspot owner j sells capacity in a single region, *i.e.*, regions do not overlap. In Section 5.4, we show how to extend our approach to support overlapping regions. As Wi-Fi may not cover the whole sector, areas without Wi-Fi coverage can be treated as special regions with no Wi-Fi bids.

Cellular resources as a Virtual Bid: Let the traffic demand vector be $D = \{d_1, d_2, ..., d_m\}$, where d_i is the demand in region i. In order to effectively leverage both third party and cellular resources, we let the service provider also participate in the auction by submitting a *virtual bid*. The virtual bid is in the form of a cost function $F(z)$, where z is the total amount of spectrum used in the entire cellular sector. Let c_i be the cellular capacity in region i, let x_j be the total capacity bought from hotspot owner j. To satisfy the cellular traffic demand d_i in each region i, we must have: $c_i + \sum_{j:f(j)=i} x_j \geq d_i$. Since different regions may have different spectrum efficiency, we denote the actual

spectrum usage in region i as c_i/e_i, where e_i is the spectrum efficiency in region i. Thus the total spectrum usage is $z = \sum_{i=1}^{m} c_i/e_i$.

We consider $F(z)$ to be a piecewise linear convex function, capturing the fact that below a certain value the cost (reflecting sunk cost [77]) is very low because the service provider has already invested in buying the spectrum and needs to keep the system running; as the cellular network becomes more loaded, the cost increases; and once it is overloaded, the cost increases sharply to capture the high cost of congestion. Thus z is not limited by the available spectrum and can go to infinity. A similar convex cost function has been widely used in modeling congestion cost in the Internet (*e.g.*, [26, 71, 70]).

Because the cellular resource in the virtual bid can be used in any region in the sector, it introduces coupling between the regions. The entire sector can now be viewed as one auction instead of several independent ones as in the naïve solution. Even if the number of hotspots in one region is small, its hotspots are not guaranteed to win since the auction may buy more Wi-Fi from other regions and save the cellular resource for this region, *i.e.*, hotspots compete not only within their regions, but also across regions. We now see a new type of competition, which we call *inter-region competition* in addition to *intra-region competition*.

Auction objective: The goal of the cellular service provider is to minimize the total Wi-Fi and cellular cost, while satisfying the customers' demands (*i.e.*, $c_i + \sum_{j:f(j)=i} x_j \geq d_i$) and offering appropriate incentives to the third-party Wi-Fi hotspot owners to share their resources.

5.2.2 Preparation: (Static) Global Allocation

We first ignore traffic variations and develop techniques to effectively utilize both cellular and Wi-Fi resources in serving user traffic demands.

$\triangleright$ *Input* : $d_i,\ e_i,\ \lambda_j,\ p_j,\ F(z)$
$\triangleright$ *Output* : $x_j,\ c_i,\ z$
minimize: $\sum_j p_j * x_j + F(z)$
subject to:

[C1]	$\sum_{j:f(j)=i} x_j + c_i = d_i$	$\forall i = 1, 2, ..., m$
[C2]	$\sum_{i=1}^{m} c_i/e_i = z$	
[C3]	$0 \leq x_j \leq \lambda_j$	$\forall j = 1, 2, ..., n$
[C4]	$0 \leq c_i$	$\forall i = 1, 2, ..., m$

Figure 5.2: Problem formulation to optimize allocation

We formulate a *global resource allocation problem* as a linear program in Figure 5.2. The formulation effectively captures global cellular resources and local Wi-Fi resources by treating the cellular resource as a single resource with a single bid. As shown, our goal is to minimize the sum of total Wi-Fi cost (based on their bids) plus cellular cost $F(z)$. The constraint [C1] ensures that we have enough Wi-Fi and cellular resources to satisfy traffic demands in each region i. The constraint [C2] relates the cellular capacity with the cellular spectrum. The constraints [C3] and [C4] put upper and lower bounds on x_j and c_i. Since there is no upper bound on z, there is always a feasible solution. When z increases beyond the available spectrum, $F(z)$ grows rapidly to reflect high congestion cost. This problem can be solved efficiently using linear program solvers, (*e.g.*, CPLEX).

5.2.3 iDEAL Dynamic Global Allocation

Traffic demand changes over time and is challenging to predict accurately. Based on the history of observed demand vectors, we can optimize for the representative demand vectors that are likely to occur in the next time interval. Our goal is to find the allocation to minimize the worst-case cost for these representative demand vectors.

Algorithm: Formally, suppose there are K historical demand vectors, denoted as $D_k = (d_{k1}, d_{k2}, \cdots, d_{km})$ $(k = 1, \cdots, K)$, where d_{ki} denotes the k-th possible demand in region i $(i = 1, \cdots, m)$. While it is difficult to predict accurately the demand vector for the next time interval, it is common in robust traffic engineering to assume that the demand vector for the next time interval is covered by the convex hull of all the historical demand vectors D_k [71]. Under this assumption, we can minimize the worst-case cost while satisfying all possible demands that may arise in the next time interval. We formulate this dynamic global allocation problem by modifying the LP formulation in Figure 5.2. In particular, we change [C1] and [C2] to the following:

$$\text{[C1-dynamic]} \qquad \sum\nolimits_{f(j)=i} x_j + c_{ki} \geq d_{ki} \quad \forall\ k \text{ and } i$$

$$\text{[C2-dynamic]} \qquad \sum\nolimits_{i} (c_{ki}/e_i) = z \quad \forall\ k$$

to ensure that we have enough cellular and Wi-Fi resources to satisfy all possible demand vectors. This is much more efficient than provisioning for the peak demand in each region.

From now on, we will refer to our dynamic global allocation algorithm as *iDEAL*, and the static global allocation algorithm as *iDEAL (static)*.

Property: A nice property of this dynamic global allocation is that it effectively leverages the global cellular resource on demand to satisfy different possible traffic demands. In particular, while the total cellular resource is fixed, the amount of cellular resource used in each region can change according to the real demand. When demand shifts from one region to another over time, the same global cellular resource can be used, instead of provisioning for the peak demand in each region. Therefore, global cellular resource has a distinctive advantage over local Wi-Fi resources in satisfying time-varying demand, which we explicitly leverage in our formulation.

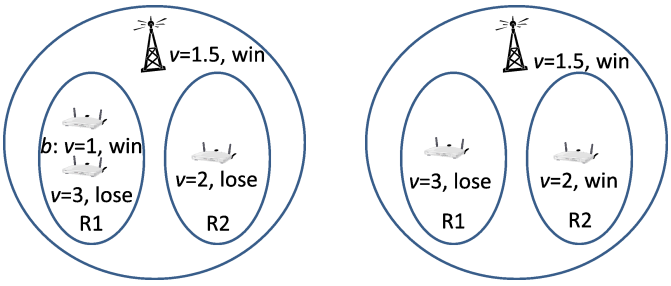
5.2.4 iDEAL Pricing Solution

As discussed in section 5.1, we want the pricing scheme to be truthful and efficient. Meanwhile, we want the pricing scheme to fully benefit from the inter-region competition. For example, when hotspots in one region lower their bids and offload more traffic, this would reduce the demand for third party resources in other regions and cause hotspots in other regions to sell less. To capture this unique interaction between intra-region and inter-region competition, we cannot treat auctions in different regions as separate auctions and compute pricing separately; instead we must consider them as a single auction and explicitly incorporate inter-region competition into the payment computation.

The Vickrey-Clarke-Groves [82] auction is well-known. It is both truthful and efficient. It pays a winner the opportunity cost that the presence of the winner introduces on the other players. VCG has a major weakness – its cost is generally high [9]. However, in our setting VCG is able to capture the inter-region competition, which lowers the cost. Thus to preserve the nice properties of VCG (*i.e.*, truthfulness and efficiency) while achieving low cost, we apply the VCG principle globally over the whole cellular sector and compute the *global opportunity cost* to capture both inter-region and intra-region competition.

Algorithm: We follow the general VCG principle and compute the global opportunity cost as follows. Let $V(D, N)$ denote the valuation consumed in the optimal allocation. D is a demand matrix containing K demand vectors $D_k = \{d_{k1}, d_{k2}, \cdots, d_{km}\}$ $(k = 1, \cdots, K)$, which specify possible demands in each region. N is the set of bidders (including the cellular service provider). Given the result of the allocation scheme, if we buy t capacity from winner b in region r, the amount of money we pay to b will be $V(D, N \backslash \{b\}) - V(D^1, N \backslash \{b\})$ where D^1 is derived from D by setting $d_{kr} = max(0, d_{kr} - t)$ for each k and $N \setminus \{b\}$ is the set of remaining bidders after removing bidder b. Thus, $V(D^1, N \setminus \{b\})$ is the total value sold by other bidders under the current optimal allocation; $V(D, N \setminus \{b\})$ is the total valuation optimized after removing b. The difference is the global opportunity cost b imposed on other bidders.

Next we use an example to show how global opportunity cost is computed and how the inter-region competition can help reduce cost. Figure 5.3



(a) Optimal allocation

(b) Optimal allocation after removing the winning hotspot

Figure 5.3: Global opportunity cost example.

shows two regions $R1$ and $R2$, each with 1 unit demand. $R1$ has 2 hotspots with valuations 1 and 3, respectively. $R2$ has 1 hotspot with valuation 2. Each hotspot has 1 unit resource. The cellular resource is 1 unit and is worth 1.5. The optimal allocation is shown in Figure 5.3(a): 1 unit of Wi-Fi in region 1 with valuation 1 and 1 unit of cellular resource in region 2. To compute the global opportunity cost for the Wi-Fi winner b, we remove b and compute the optimal allocation without the winner as shown in Figure 5.3(b). The new allocation should use all the cellular resource in region 1 and the Wi-Fi resource with valuation 2 in region 2. The total valuation sold by other bidders is thus $1.5 + 2 = 3.5$, while in the original allocation the number is 1.5. So the global opportunity cost we pay to b is $3.5 - 1.5 = 2$. In comparison, with the same allocation, if we apply VCG in each region separately, the local opportunity cost is 3 since region 1 has only the Wi-Fi resource with valuation of 3 after we remove b. This shows that global opportunity cost is lower since it effectively takes into account resources across all regions. Note that this notion of global opportunity cost and its computation work for both static and dynamic

global allocations. The two versions only differ in the allocation as described in Section 5.2.2 and 5.2.3.

Properties: iDEAL inherits the following three important properties from VCG: (i) bidders have incentives to be *truthful*, (ii) the outcome of the auction is *efficient*, and (iii) the auction is *individually rational* meaning third-party resource owners have incentives to participate in the auction. Formally, we have the following three theorems.

Theorem 4. *In iDEAL, truth-telling is an optimal strategy.*

Proof. Pick arbitrary bidder b_1 in region r_1, let its valuation for unit capacity (e.g. 1 bps) be v_1. Suppose when b_1 bids truthfully, it sells t_1 amount of capacity. Its utility is

$$U1 = (V(D, N \setminus \{b_1\}) - V(D^1, N \setminus \{b_1\})) - v_1 \cdot t_1,$$

i.e., the difference between the payment it receives and its valuation for the amount it sells, where D^1 is derived from D by setting $d_{kr_1} = max(0, d_{kr_1} - t_1)$ for each snapshot k.

If b_1 bids untruthfully and sells t_2 amount of capacity, its utility is: $U2 = (V(D, N \setminus \{b_1\}) - V(D^2, N \setminus \{b_1\})) - v_1 \cdot t_2$, where D^2 is derived from D by setting $d_{kr_1} = max(0, d_{kr_1} - t_2)$ for each k.

In order to prove truth telling is an optimal strategy we need $U1 \geq U2$.

It is evident that:

$$\begin{aligned} U1 - U2 = \ & [v_1 \cdot t_2 + V(D^2, N \setminus \{b_1\})] - \\ & [v_1 \cdot t_1 + V(D^1, N \setminus \{b_1\})]. \end{aligned}$$

Here the first term is the minimum total valuation needed when we buy t_2 from b_1 and the second term is the total valuation used in the optimal allocation. Since the second term is optimal, the first term cannot be smaller and thus $U1 \geq U2$. □

Theorem 5. *iDEAL is efficient, which means when bidders are rational, the winners are the bidders whose valuation for their resources is the least.*

Theorem 6. *iDEAL is individually rational,* i.e.*, bidders of the auction will get non-negative utility, assuming a bidder does not bid lower than his valuation.*

Theorem 4 indicates that it is beneficial for a bidder to bid truthfully regardless of other bidders' strategies. Theorem 5 follows from the truthfulness property and our allocation, which minimizes the total valuation assuming everyone bids truthfully. Theorem 6 guarantees that winners will be paid no less than their valuation.

While Theorem 6 is easy to see in normal settings, it is less straightforward with our dynamic allocation because in the dynamic allocation the total amount of resource we buy is not fixed. Specifically, when computing the opportunity cost, we remove a winner and compute a new allocation and

use the bid(s) of the newly admitted winner(s) as the payment. While the unit prices of the newly admitted bids are not lower than the winner's, the total amount of capacity we buy in the new allocation might reduce. This is because the new allocation may buy more cellular resource, which can be used everywhere and may reduce the need for Wi-Fi in all regions. That makes it hard to tell if the opportunity cost is higher than the winner's valuation. We prove the theorem using contradiction: if we remove a winner w and the amount of increased valuation we buy from others (*i.e.*, the opportunity cost) is less than what w sells, then w should not have won.

5.3 Understand and Guard Against Collusion

In this section, we first identify potential collusion strategies in iDEAL and show how they differ from those in normal VCG settings. We then discuss how to mitigate such strategies. We call a set of hotspots colluding together as a *bidding ring*. A bidding ring colludes by adopting a certain bidding strategy to maximize utility, *i.e.*, the difference between the payment and the true valuation of the resource sold.

5.3.1 Collusion Strategies

Due to the distributed nature of hotspot locations, collusion in our context is quite different from collusion in normal settings, where the optimal collusion strategy is to let one proxy bidder buy (or sell in an reverse auction) for the whole bidding ring [10]. However, in our system each hotspot submits

a separate bid. This forbids hotspots to collude optimally and thus may resort to other collusion strategies identified below. In particular, we consider two types of collusion: (i) single seller collusion, whose objective is to maximize the total utility of all hotspots owned by this seller, and (ii) multi-seller collusion, where each seller colludes with other sellers, but tries to solely maximize its own utility.

In both types of collusion, a bidding ring can drive up the price and increase its utility by *Supply Reduction* (*i.e.*, drop losing bids or reduce the capacity offered in winning bids, which is equivalent to bidding an extremely high price for the capacity that is removed from bidding). Supply reduction can drive up price because it increases the opportunity cost, which is determined by the immediate losing bids.

5.3.2 Mitigating Collusion

We mitigate collusion as follows:

Bidding as a group to address single-seller collusion: A single seller with multiple hotspots has an incentive to reduce supply because its hotspots submit separate bids. The opportunity cost of one hotspot can be affected by the price/availability of its other hotspots. So by strategically dropping some of its hotspots or raising their prices, it can increase its revenue. This strategy is especially harmful as it may also increase the opportunity cost of other sellers' hotspots. Ultimately, it incurs a higher cost to the service provider.

To address the issue, we let the hotspots owned by the same entity bid

as a group, *i.e.*, the seller who owns multiple hotspots discloses all its hotspots and we consider them as a single bidder in the auction. The seller has an incentive to choose this option, since bidding truthfully is an optimal strategy (Theorem 1). It is also preferred from the service provider's perspective because it only removes competition within the group. The hotspots in this group still compete with hotspots of other sellers, which helps to bring down the cost.

Dynamic demands in multi-seller collusion: In order to benefit from supply reduction, a bidding ring needs to accurately predict which bids may lose and drop them. Without that, supply reduction can cause harm by letting the bidding ring miss opportunities to win. Making such predictions is challenging due to the dynamic nature of the traffic demand and Wi-Fi availability. Therefore, in practice supply reduction does not necessarily increase the utility of the hotspots, which can discourage them from colluding.

Stability of multi-seller collusion: When multiple parties are involved in collusion, a natural question is whether the collusion is stable (*i.e.*, all members of the bidding ring have incentives to stay in the ring [10, 17]). [10] shows that in normal settings, collusion in VCG is stable under certain assumptions. However, their conclusion does not apply to our context because of the difference in collusion strategies. Specifically, we make the following two observations.

First, without utility sharing, members of a bidding ring have an in-

centive to leave the ring (*i.e.*, do not conduct supply reduction). Formally, we have the following lemma:

Lemma 7. *Without utility sharing, for bidding ring members no supply reduction is a (weakly) dominant strategy (*i.e.*, no worse than supply reduction).*

This follows from the truthfulness of VCG and the fact that different sellers submit separate, sealed bids and cannot pose as one entity in our system.

Second, the condition of "no utility sharing" is likely to hold in practice due to difficulties of estimating utility obtained from collusion in our system. One reason is that traffic demands and Wi-Fi availabilities are highly dynamic, which makes it hard to attribute utility changes to collusion. Moreover, using sealed bids makes it hard to validate the behavior of other members in the bidding ring. We can make it even harder through system design such as delayed payment (*e.g.*, paying the hotspots every week even though the auction is conducted hourly), which further obfuscates the utility.

5.4 Practical Considerations

Allowing general bidding curves: Section 5.2 assumes the cellular cost $F(z)$ is convex, and hotspot owners can bid only one price value p_j for the entire capacity they offer and are willing to sell part of the capacity at a prorated amount. Now we consider more general cost functions, including convex, concave (capturing economy of scale), or a combination. For example, a third-party hotspot owner may want the flexibility of specifying an entire bidding

curve characterizing the ask price with respect to the amount of offloaded traffic. The formulation shown in Figure 5.2 remains the same, except that the optimization is no longer a linear program or convex program due to the more flexible p_j and $F(z)$.

When only $F(z)$ is non-convex and hotspot owners still bid only one price value for the entire capacity they offer, the problem is easy to solve. We approximate $F(z)$ as a piecewise linear function and solve the optimization problem by enumerating all possible line segments that z belongs to (because within each segment $F(z)$ is still a linear function). Essentially we solve a linear programming problem for each possible line segment, and then pick the best result from all the corresponding linear programs.

To support general Wi-Fi hotspot cost functions, we use dynamic programming. We introduce discrete step sizes s and s' that quantify the smallest cellular and Wi-Fi capacity unit to purchase, respectively. Small step sizes give better solution but increases running time. We build a table T, where each entry $T(k, z)$ gives the cost of satisfying the demands of the first k regions: $1, 2, ..., k$ using cellular capacity equal to z.

$$T(k,z) = \min_{v}\{T(k-1, z-v) + F(z) - F(z-v) \\ + auctionCost(k, d_k - v \cdot e_k)\},$$

where $auctionCost(k, x)$ is the minimum cost for satisfying x amount of demand using Wi-Fi in region k, d_k is the demand in region k, e_k is the spectral efficiency in region k, both z and v are multiples of step size s.

We compute *auctionCost* using Dynamic Programming as follows:

$$W_k(i, y) = \min_u \{W_k(i-1, y-u) + cost(i, u)\},$$

W_k is the Wi-Fi cost table for region k. Here $i = 1, 2, ...N_k$ where N_k is the number of bidders in region k. The amount of capacity we seek to satisfy, y, varies from 0 to x, where x is the total amount of capacity we want to satisfy with Wi-Fi in region k. $cost(i, u)$ is simply the cost of satisfying u amount of demand using only bidder i at his bid price. The expression indicates the cost of using i hotspots to satisfy y demand in region k is the minimum cost of using $i-1$ hotspots to satisfy $y-u$ demand plus the cost of using i-th hotspot to serve u demand. y and u are multiples of step size s'. We then have $auctionCost(k, x) = W_k(N_k, x)$.

Supporting offloading to femtocells and dynamic roaming: In addition to third-party Wi-Fi hotspots, femtocells and other cellular networks can also be used for offloading. Roaming to other cellular networks considered here is different from traditional roaming. Traditional roaming is enabled only outside the current cellular provider's coverage area whereas dynamic roaming in our context can take place within the coverage area to reduce congestion. In order to support offloading to different types of technologies, we need to effectively handle partially overlapping spatial coverage, as different resources have different coverage ranges. This requires changes to the allocation algorithm. We extend our approach to support these scenarios by dividing overlapping regions into multiple non-overlapping regions and allowing one provider to be-

long to multiple regions. The constraint [C1] in Figure 5.2 is then replaced by the following two new constraints:

$$\sum\nolimits_{j:i\in f(j)} x_{ji} + c_i = d_i, \qquad \forall i = 1, 2, ..., m,$$
$$\sum\nolimits_{i} x_{ji} = x_j, \qquad \forall j = 1, 2, ..., n,$$

where x_{ji} is the amount of capacity bought from seller j and used in region i. This extension can not only support offloading to different types of networks, but also allow a hotspot provider to use its resources across different regions (*e.g.*, hotspots belonging to a single restaurant chain spread across different regions but sharing the same bottleneck capacity).

Incorporating quality score: The cellular service provider may prefer some hotspots over others due to different quality (*e.g.*, to avoid hotspots that do not guarantee the amount of capacity they offer). In this case, we can differentiate which hotspots to use based on the quality score q_i ($0 < q_i \leq 1$) of hotspot i. The higher the score, the better the quality and the easier the hotspot can win in future auctions. To achieve that and ensure the auction is still truthful and individually rational, we change the objective function in the allocation phase to $\sum_j (x_j \cdot p_j/q_j) + F(z)$, which essentially increases the bid of hotspots with low quality scores and makes it harder for them to win. We also change the payment for winner j to q_j times the opportunity cost so that individual rationality is still preserved because the opportunity cost is no less than $x_j \cdot p_j/q_j$. It is not difficult to see that the auction is still truthful, since the quality scores are bid-independent.

Benefiting from delay-tolerant demands: Some application traffic (*e.g.*, emails) is delay tolerant. A natural way to take advantage of such traffic is to delay them when it is too costly to satisfy them immediately (*e.g.*, when the current traffic load is very heavy or when most traffic demands originate from outside the Wi-Fi coverage areas and have to be satisfied by only the cellular network).

Our framework is flexible enough to support this new optimization task. Consider traffic demands in m snapshots, namely $D = \{d_{k,i}, d_{2,i}, ..., d_{k,i}...\}$. We can optionally delay a certain demand in snapshot i to snapshot j, where $i < j$. The resulting demand (called final demand) becomes $D' = \{d_1 - \delta_1, d_2 - \delta_2 + \delta_{1,2}, ..., d_i - \delta_i + \sum_{j<i} \delta_{j,i}, ..., d_k + \sum_{j<m} \delta_{j,m}\}$, where δ_i denotes the total amount of traffic in snapshot i delayed to future snapshots and $\delta_{i,j}$ denotes the amount of traffic in snapshot i delayed to snapshot j. It is easy to see $\delta_i = \sum_{j>i} \delta_{i,j}$. Our goal is to satisfy D' during every interval i while minimizing the total cost of satisfying the demands over all intervals plus the penalty incurred in delaying traffic. This can be formulated as an LP problem shown in Figure 5.4. The variables are defined in Table 5.1 except that now some variables have the additional subscript k to denote their values in snapshot k. The objective reflects the cost of satisfying the final traffic demand in all snapshots plus the penalty associated with delaying traffic. [C1] enforces that the final traffic demand is satisfied during every snapshot. [C2] captures total cellular spectrum usage in snapshot k. [C3] and [C4] provide bounds on $x_{k,j}$ and $c_{k,i}$. This extension changes the allocation algorithm in

iDEAL, but the same pricing algorithms and proofs in Section 5.2.4 still apply.

$\triangleright$ *Input* : $d_{k,i},\ e_i,\ \lambda_{k,j},\ p_{k,j},\ F(z)$
$\triangleright$ *Output* : $x_{k,j},\ c_{k,i},\ z_k, \delta_i, \delta_{j,i}$
minimize: $\sum_k (\sum_j p_{k,j} x_{k,j} + F(z_k)) + penalty \sum_{k>j} (k-j)\delta_{j,k}$
subject to:

[C1]	$\sum_{j:f(j)=i} x_{k,j} + c_{k,i} = d_{k,i} - \delta_{k,i} + \sum_{j<k} \delta_{j,k,i}$	$\forall k, i$
[C2]	$\sum_i c_{k,i}/e_i = z_k$	$\forall k, i$
[C3]	$0 \le x_{k,j} \le \lambda_{k,j}$	$\forall k, j$
[C4]	$0 \le c_{k,i}$	$\forall k, i$

Figure 5.4: Modified problem formulation to optimize allocation.

Selecting users to offload: So far, we have considered how much capacity to buy from each Wi-Fi hotspot. Now we study which users should be offloaded to a particular Wi-Fi hotspot. We prefer a scheme that minimizes the switching time (*e.g.*, avoid offloading users who will soon leave the hotspot or have little traffic to send), while fully utilizing the purchased capacity at the hotspot. We analyze the Wi-Fi traces from the large cellular provider in US, which also provides Wi-Fi services, and find that a user who has stayed at a hotspot for a longer time in the past is more likely to stay longer in the future. Therefore, we propose a simple heuristic, which is to offload a user to the hotspot if (i) the user has already stayed there for at least a threshold amount of time (*e.g.*, 5 seconds), and (ii) his estimated traffic demand is lower than the residual purchased capacity at the hotspot. The condition (ii) tries to avoid switching the user back and forth between the Wi-Fi hotspot and the cellular network. Thus, users who are likely to soon leave the hotspot do not have to incur the

Wi-Fi switching overhead.

Supporting unsplittable demand: A single device usually can only connect to one hotspot at a time. To capture such unsplittable demand, we add the following constraint to the formulation in Figure 5.2: the demand of each device has an indicator t_{kj}, where $t_{kj} = 1$ when demand b_k is assigned to hotspot j and 0 otherwise. Therefore, for each demand b_k, $\sum_j t_{kj} \leq 1$. For each hotspot j, $\sum_k b_k \times t_{kj} \leq x_j$. To solve this integer programming, we can first relax integer constraint on t_{kj} and solve the relaxed LP. Then we round t_{kj} to 0 or 1. Similar rounding approach has been successfully used to find an approximate solution to several integer programming problems (*e.g.*, [74]).

Chapter 6

iDEAL Evaluation

6.1 Evaluation Methodology

We evaluate our approach using trace-driven simulations. We first describe the traces and how they are used.

Traces: We use the following traces: (i) Locations of cell towers and femtocells from a large cellular provider in the US, and hotspot locations from [86]. (ii) Detailed network data with periodic (every 2 seconds) reports of which sectors mobile devices are using for their data communication. We use one-week data from 2011, and pick the busiest sectors out of thousands of sectors. We then use this data to estimate the number of users in a sector during one hour, and the amount of time they stay in that sector. (iii) 3G HTTP traces report detailed HTTP session information, such as HTTP duration, downloaded bytes and type of the download during all 24 hours on a single day in 2011. This is aggregated over several sectors and does not have information about which sector the user currently is in. (iv) The backhaul capacity of about 150 hotspots from a large service provider in the US.

Generating regions: We generate regions by clustering the Wi-Fi hotspots using k-means [51]. We use 6 regions as it minimizes partition index [15], which

is a desirable clustering goal. We run the clustering algorithm 100 times and pick the clustering that minimizes the average distance of Wi-Fi hotspots to the centers of their assigned regions.

Network configuration: Based on the typical cell tower spacing of 400-500m in busy urban areas, we use 250m as the communication range for a cell sector. The communication ranges for Wi-Fi and femtocell are set to 100 m and 40 m [8], respectively. To calculate spectrum efficiency, we use the distance between the centroid of the region and the cell tower, and the distance between the centroid of the region and the interfering cell towers, and compute path loss using Hata model [33]. We consider 6 nearest base-stations as interfering base-stations to calculate the $SINR$. We account for self-interference and compute the resulting $SINR'$ as: $SINR' = \frac{SINR}{1+\alpha * SINR}$ where $SINR'$ and $SINR$ denote the signal to interference and noise ratios with and without self interference, respectively, and $\alpha = 0.005$ [4]. We get the spectrum efficiency by applying Shannon's Law. Since the Shannon capacity is an over-estimate of the real capacity, we scale down the result to match the maximum efficiency that is generally observed in a cellular network (2 bps/Hz).

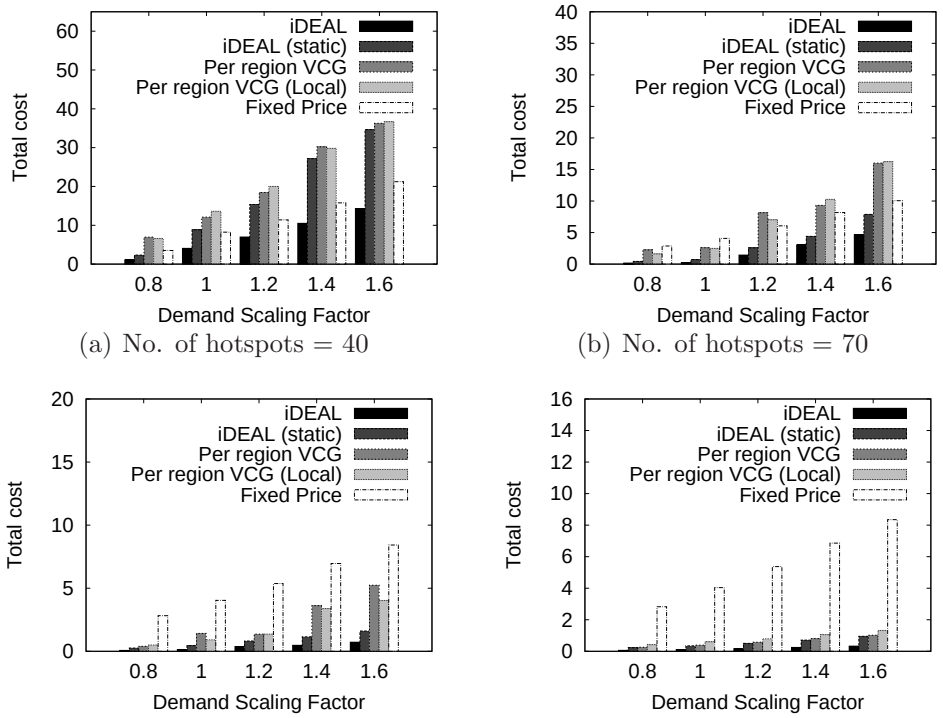
Generating traffic demands: To generate the demand for an hour, we determine the number of users from the detailed network data during that hour, and pick all the HTTP requests of the corresponding number of users picked from the 3G HTTP trace. We replace the data rate in the trace with the desired demand rate according to the application types: video 350 kbps, audio 128 kbps, application (*e.g.*, download binary files) 350 kbps, text 150

kbps, and image 165 kbps. We determine the rates of applications, text, and images according to the 90-th percentile rate that users receive from the 3G HTTP trace, and determine the video and audio rates using the data from a large service provider. The data rates in the traces are not used since they are limited by the current cellular capacity and may not indicate the real demand.

We place users randomly in the sector and assign them to regions according to their locations. When a single demand vector is used, we use the peak demand from each region as the final traffic demand. When dynamic allocation is used, we use all the demand vectors corresponding to the time when any region has peak demand. This way, both static and dynamic allocation schemes can sustain the peak loads in all regions.

Generating bids: We use the distribution of backhaul data-rates and pick the available data rate uniformly as being 25%-75% of the backhaul data-rate. The Wi-Fi bids are then generated based on the pricing plan of a major service provider. We uniformly choose 50%-150% of the price as a hotspot's valuation for a given backhaul capacity to capture varying costs from different service providers. We then determine the hourly Wi-Fi valuations according to its capacity and monthly bills assuming 30 days/month and 8 hours/day. The real bids depend on their bidding strategies and may differ from their valuations. The cellular bid $F(z)$ is set to 0 (reflecting the sunk costs) when z is below 80% of cellular capacity (which is set to 3 carriers *i.e.*, 3 times 3.84 MHz), and set to c times estimated maximum Wi-Fi valuation when z exceeds 80%. We set c to 1.25 by default and vary it, to evaluate its impact.

Performance metrics: We compare different schemes using efficiency and cost. Efficiency is measured as the total valuation of all resources consumed, whereas cost is the total price of cellular and Wi-Fi resources the service provider pays.



(a) No. of hotspots = 40

(b) No. of hotspots = 70

(c) No. of hotspots = 100

(d) No. of hotspots = 130

Figure 6.1: Total cost comparison with truthful bids

6.2 Evaluation Results

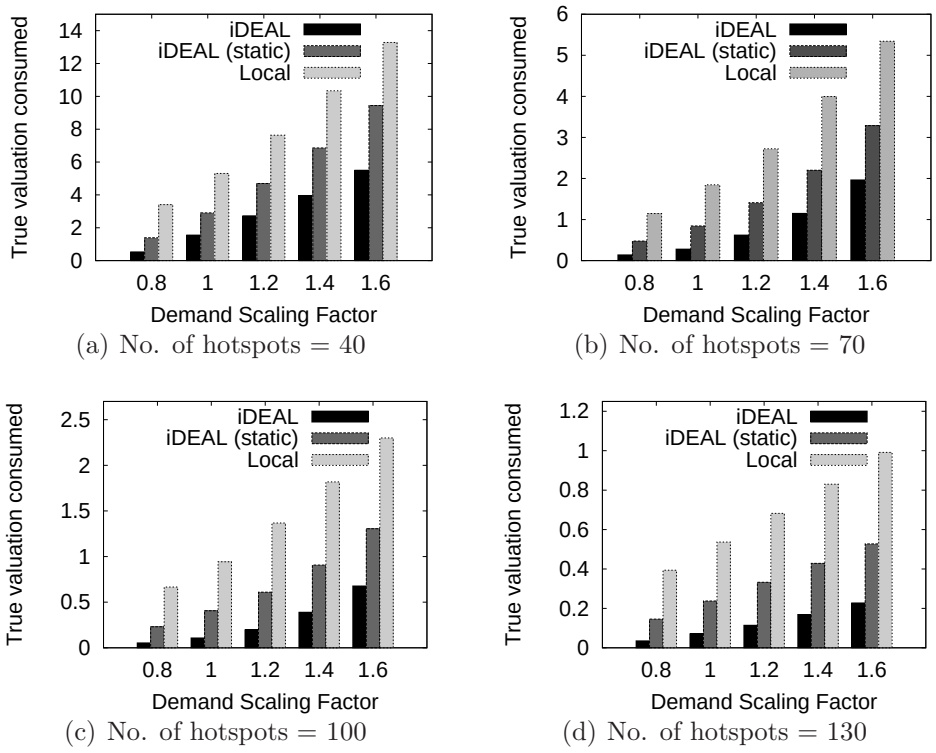
6.2.1 Comparison of Truthful Auctions

We first compare the cost incurred under different auction schemes, including iDEAL, iDEAL (static), per region VCG with global allocation, and per region VCG with local allocation. All the auctions are truthful except per

region VCG with global allocation, which is included to show how VCG will perform without inter-region competition. In addition, we also compare with fixed pricing, where the service provider pays the hotspots a fixed price and uses the global allocation to determine which hotspots to buy. A hotspot with higher valuation than the fixed price would not sell in this case, so we use the maximum Wi-Fi price we may generate as the fixed price. The result of using average Wi-Fi price as the fixed price is similar and omitted for brevity.

Figure 6.1 shows the cost incurred under different schemes. We first observe that auction based approaches work much better than the fixed pricing when there is enough competition. When the number of hotspots is small, fixed pricing can perform better than most auction based approaches. However, iDEAL achieves lower cost than the fixed pricing even when the number of hotspots is 40. With 130 hotspots, iDEAL is almost an order of magnitude better than the fixed pricing. Second, iDEAL out-performs iDEAL (static), which out-performs both versions of per region VCG. Per region VCG fails to capture the inter-region competition and thus may suffer from limited competition and lead to high cost. In comparison, both versions of iDEAL fully benefit from inter-region competition. iDEAL further reduces cost by leveraging the flexibility of using cellular resource in different regions on demand, thus reducing the demands for third party resources. Therefore, iDEAL and iDEAL (static) out-perform per region VCG by 63%-80% and 10%-61%, respectively.

We further compare the efficiency of the following allocations, all with truthful bids: (i) iDEAL, which can optimize allocation according to multiple



(a) No. of hotspots = 40 (b) No. of hotspots = 70

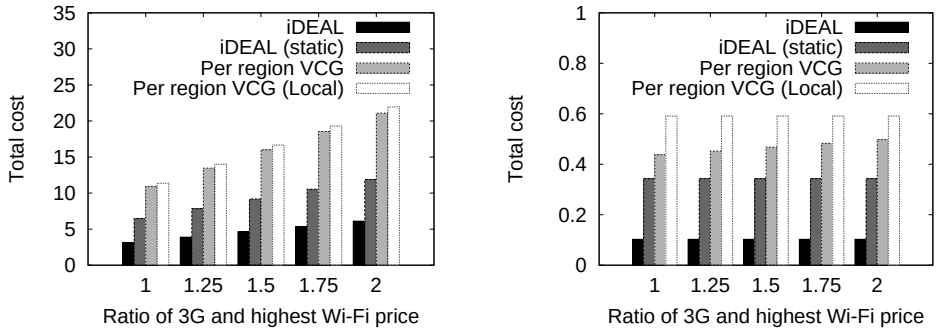
(c) No. of hotspots = 100 (d) No. of hotspots = 130

Figure 6.2: Comparison of total true valuation consumed.

possible demands, (ii) iDEAL (static), which optimizes allocation according to a single traffic demand, (iii) local allocation, which statically allocates cellular resources to different regions based on the traffic demands in these regions. Note here we omit the fixed pricing because it is not an auction and it makes allocation decisions solely based on the fixed price instead of the valuation. Figure 6.2 shows the total true valuation of different allocation schemes as we scale the traffic demands by a constant factor from 0.8 to 1.6 and vary the total number of hotspots participating in the auction. As before, iDEAL outperforms its static counterpart, iDEAL (static), which further out-performs the local allocation. iDEAL reduces the total valuation to only 8%-42% of

local allocation since it can effectively adapt the cellular allocation to different regions based on real demand. Even iDEAL (static) performs very well: its total valuation consumed is only 34%-72% of local allocation.

Figure 6.3 further compares the cost of different auction schemes as we vary the cellular cost $F(z)$ by changing its parameter c from 1 to 2, where the cellular bid is set to c times estimated maximum Wi-Fi valuation when z exceeds 80%. The absolute cost increases with c, as we would expect. The relative performance across different schemes is similar for all values of c we use. The total cost reduces as competition increases (*i.e.*, when the number of hotspots goes up from 40 to 130).



(a) No. of hotspots = 40 (b) No. of hotspots = 130

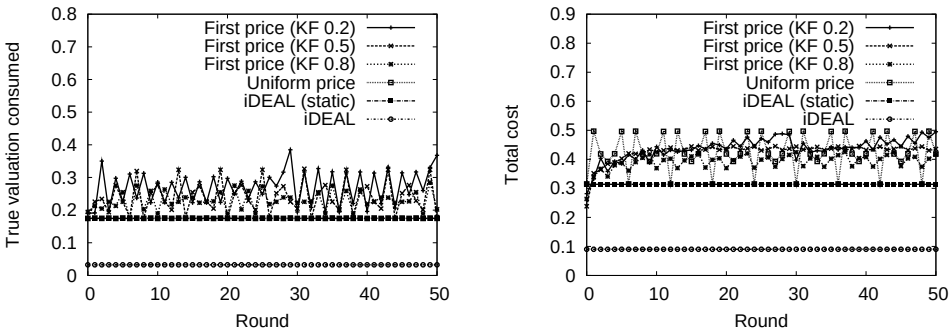
Figure 6.3: Total cost comparison with varying cellular cost function

6.2.2 Comparison with Non-truthful Auctions

In this section, we study the impact of individual hotspot gaming in non-truthful auctions. We compare iDEAL with the first price and regional uniform price, both of which are widely used [21, 44]. The first price pays winners the amount of their bids, and the regional uniform price pays all the winners in a region at the first losing bid in the region. We do not compare with

GSP (Generalized second-price auction) because unlike GSP, iDEAL does not differentiate between winning slots. If GSP were used, everyone would game to be the highest paid winner as in the first price. We use the static global allocation for all schemes, except that iDEAL uses dynamic global allocation. There are many possible gaming strategies. In our evaluation, we consider simple gaming strategies as examples and show that even these simple strategies can significantly degrade performance. In the first price, we assume a bidder can observe some fraction of bids from other bidders in his region. We call this fraction as Knowledge Factor (KF). He then uses that information to guide his bid in the next round by bidding the maximum among (i) his valuation, (ii) the average of the lowest losing price he sees, and (iii) the highest winning price he sees (including his own bid in the last round). In the first round, bidders start by bidding uniformly randomly between one time and two times their valuation. In the uniform price auction, bidders can game by supply reduction. So we let the winners who do not sell all their capacity reduce their capacity to slightly below the amount they sell in the hope of admitting new winners and potentially increasing the price. When they do sell all their capacity, they will try to increase their offered capacity. In reality, bidders can be more aggressive. For example, all bidders may attempt to reduce supply (*e.g.*, even when they sell all they offer, they can potentially still gain by supply reduction), which may harm the system even further. We conduct multiple runs, and show the results from one run since they are all similar.

Figure 6.4 (a) shows how gaming affects efficiency. We make a few



(a) True valuation consumed (b) Cost

Figure 6.4: Cost of gaming

observations. First, both versions of iDEAL consume less total valuation. The total valuation of iDEAL is as low as 8% of the first price due to more effective use of cellular resources in presence of multiple demands. The total valuation of iDEAL (static) is only 45% of the first price. Second, both versions of iDEAL are stable as bidders are truthful. In comparison, the total value consumption fluctuates considerably in the first price auction because the bidders adapt their bids according to the others' bids. The uniform price performs close to iDEAL (static), because the bidders in our simulation only reduce supply slightly and they do not game by asking higher. In reality, the damage can only be worse.

Figure 6.4 (b) further compares the total cost to the provider. Similar to the case of total valuation, both iDEAL versions yield significantly lower cost. Specifically, iDEAL reduces the cost to 18% of the first price and regional uniform price. Moreover, even iDEAL (static) reduces the total cost to 63% of first price and regional uniform price. This result shows that with the help of inter-region competition, using VCG does not incur higher cost than first

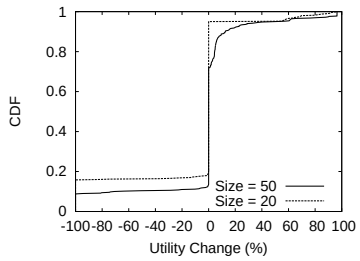
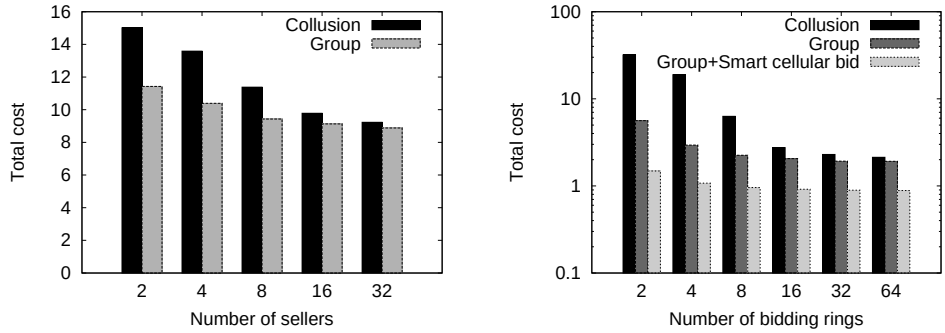


Figure 6.5: CDF of Utility Change due to Collusion.

price or regional uniform price.



(a) No. of hotspots = 40 (b) No. of hotspots = 70

(c) No. of hotspots = 100 (d) No. of hotspots = 130

Figure 6.6: Auction cost under collusion and with group option

6.2.3 Collusion

Collusion under dynamic demands: We first study how often a bidding ring can improve its utility by supply reduction. We use two different sizes of

the bidding rings: 20 and 50 out of 144 hotspots. For each size, we run the experiment 10 times with different random sets of hotspots. Each run consists of 50 rounds. In each round, the bidding ring drops all losing hotspots from the previous round. If there is no losing hotspot, it brings back the cheapest previously dropped hotspot. We vary the demand during each round, but keep Wi-Fi bids constant. We confirm the degree of traffic variation in the hourly traffic traces in multiple cellular sectors from a major cellular provider is comparable to the traces used for our evaluation. Figure 6.5 plots the CDF of percentage of utility change of the bidding ring due to collusion. We find that for the bidding ring of size 50, collusion reduces the hotspots' utility for 13% of time and improves the utility for 28% of the time. For the bidding ring of size 20, the numbers are 20% and only 5%, respectively. When collusion reduces utility, it reduces by 79% on average, while the number for improvement is only 30%. These results suggest dynamic demand significantly reduces the incentive to collude. In reality, when Wi-Fi bids are also dynamic, it is even harder to predict which set of hotspots will lose.

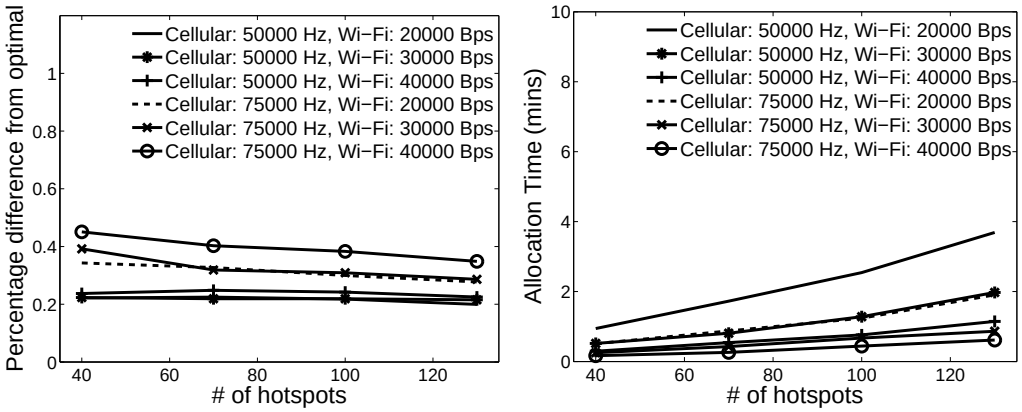
Bidding as a group: Next we compare bidding as a group with collusion using the same strategy mentioned above. Figure 6.6 plots the average cost as we vary the total number of sellers and the total number of hotspots they own and perform 100 random runs for each configuration, where each configuration generates 10 sets of sellers and 10 sets of hotspots. The results are consistent with our expectation: a single seller collusion does not always improve utility, but it always incurs a higher cost to the service provider, especially when each

seller has a large number of hotspots. In comparison, the group option, which is preferred by sellers, reduces the total cost by as much as 36% and 96% when the number of hotspots is 40 and 130, respectively. The damage of collusion reduces as the number of sellers increases since increasing the number of sellers means each seller controls fewer hotspots.

6.2.4 Extensions

Allowing bidding curves: If only $F(z)$ is non-convex, we can approximate $F(z)$ using t linear segments and optimize allocation by solving t LPs, one corresponding to each line segment. The running time increases by a factor of t. When the Wi-Fi bids are non-convex functions, we need to use the dynamic programming (DP) formulation in Section 5.4 to optimize allocation.

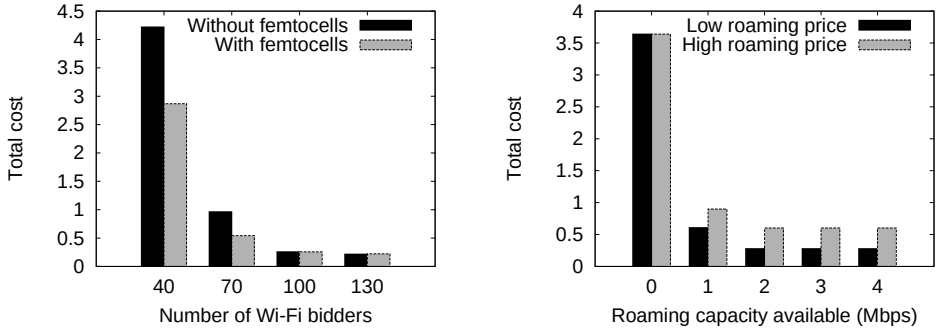
To quantify the computation cost and quality of DP solutions, we compare them with those of the LP when the Wi-Fi bids are convex (since LP can only handle convex functions). iDEAL static allocation is used in all cases. We performed the computation on a 7 core Intel(R) Xeon(R) 2.83 GHz CPU, with 32 GB RAM. Each result is an average of 5 runs. Figure 6.7(a) shows that DP increases cost by 19.9-45.1% compared to the LP due to discretization. As one would expect, smaller step sizes in the DP (defined in Section 5.4) yield closer results to the LP. This is achieved at the cost of increasing running time. As shown in Figure 6.7(b), step sizes of 50 KHz for cellular spectrum and 40 KBps for Wi-Fi capacity achieve close-to-optimal solution and take around 1 min for 130 bidders, which is affordable in practice.



(a) Cost (b) Allocation time

Figure 6.7: Performance of dynamic programming.

Supporting femtocell offload: In Figure 6.8 (a), we let both Wi-Fi hotspots and femtocells participate in the auction. We vary the number of Wi-Fi bidders while keeping 16 femtocells. As expected, the benefit of femtocells is larger when we have fewer Wi-Fi hotspots. For example, the femtocells reduce the cost by 32% when there are only 40 hotspots. As the number of hotspots increases, the additional benefit from femtocells becomes marginal since Wi-Fi has a higher communication range and is more effective in offloading.



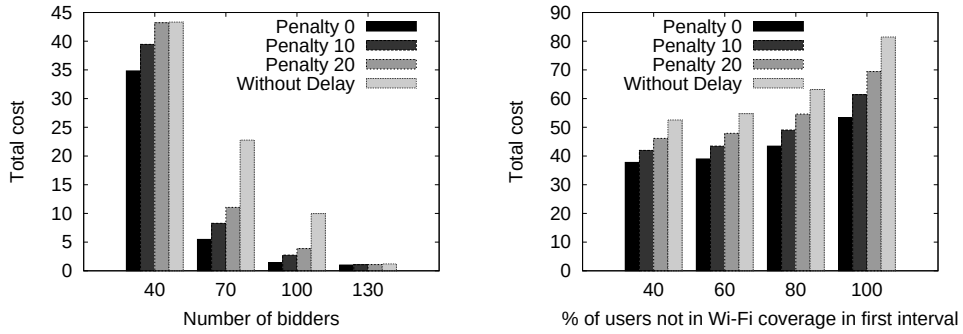
(a) Femtocell offloading (b) Roaming

Figure 6.8: Benefit of femtocell offloading and roaming

Supporting Dynamic Roaming: Figure 6.8 (b) shows the total cost as the roaming capacity available varies from 0 to 4 Mbps, where 0 corresponds to no roaming. The evaluation has 40 hotspots. In this case, since the Wi-Fi resource is insufficient, even having 1 Mbps of available roaming capacity (around 10% total cellular traffic in the sector) can significantly cut down cost. Dynamic roaming reduces the cost to 17% of that when only Wi-Fi is used with the low roaming price (which is set to the maximum winning Wi-Fi bid we observe in the default settings), and 25% under the high roaming price (which is the maximum Wi-Fi bid we may generate based on the distribution we use). Further increasing roaming capacity leads to an even lower cost but the improvement tapers off as the capacity increases beyond 2 Mbps.

Delaying delay-tolerant demands: We use two scenarios to demonstrate the benefit of delaying some traffic demands. Figure 6.9(a) shows the the total cost to serve both rounds under the first scenario, where we pick the highest demand of all the hours as the demand in the first round and use the average demand in an hour in the second round. We vary the penalty factor from 0 to infinity, where infinity means no demand will be delayed. We find delaying some demands to the second round is beneficial in all cases when penalty is smaller than infinity. The benefit is largest when the resource we have is not sufficient for the first round but sufficient for the second round. As the figure shows, the savings are 63.6% and 72.6% for 70 and 100 hotspots, respectively, when penalty factor is 10. The corresponding numbers are 51.4% and 61.2% when the penalty factor is 20.

In Figure 6.9(b), we consider the scenario where in the first round many users are not in Wi-Fi coverage and we vary the fraction of such users. In this evaluation, we only use hotspots from 3 regions and leave the other 3 regions not covered by Wi-Fi. In the second round the users are placed uniformly across the sector. The results show that delaying some demands in this case yields significant saving. When 100% of the users are not in Wi-Fi coverage in the first round, the saving can be up to 24% under penalty factor 10, and 14.8% under penalty factor 20. When 40% of the users are outside Wi-Fi coverage, the numbers become 20.2% and 12.2%, respectively.



(a) Scenario 1 (b) Scenario 2

Figure 6.9: Benefit of delay tolerant demand.

Selecting users to offload: We use one hour of the HTTP trace in one region, and determine the total Wi-Fi capacity to buy using iDEAL. We vary the Wi-Fi switching time from 0.5 seconds to 4 seconds. Figure 6.10 plots the ratio of total switching time and total offloaded time as we vary the threshold duration we use to offload a user to Wi-Fi (*e.g.*, the user has to stay at the Wi-Fi hotspot for a period that is over the threshold time). In all cases, we

ensure the Wi-Fi capacity that was purchased is fully utilized since most of the traffic come from the users who stayed at Wi-Fi much longer than the threshold time. As we can see, picking users who stay at Wi-Fi hotspot longer reduces the fraction of time spent in switching from 25.1% to 15.9% when using 4 second switching time and 5 second duration threshold.

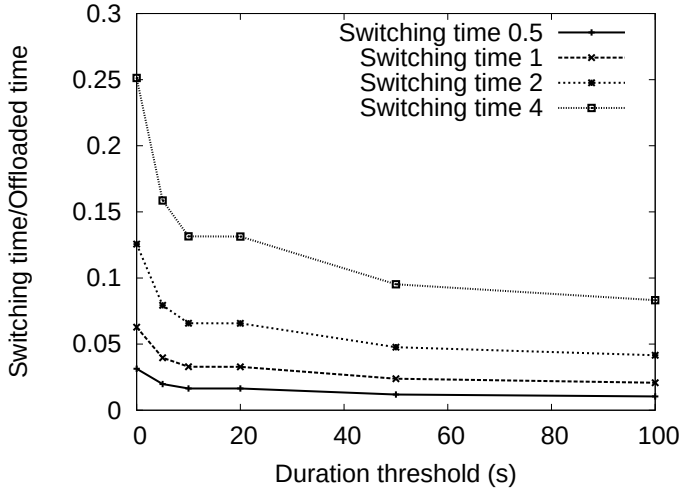


Figure 6.10: Strategically selecting users to offload reduces the switching cost.

Chapter 7

iDEAL Implementation

We describe how to offload in state-of-the-art commercial systems, and then present our implementation.

Offloading involves the following three issues: (i) identifying a network to offload, (ii) automatic authentication, and (iii) seamless offload so that the existing sessions are maintained during the offload. iDEAL already solves the first issue. Here, we describe the other two issues.

Automatic authentication: 3GPP Release 7 uses the Extensible Authentication Protocol (EAP) for key distribution to WLANs that are owned by the cellular service provider. Hotspot 2.0 is developed to support authentication with externally owned hotspots. Specifically, Hotspot 2.0 uses 802.11u to support the Access Network Query Protocol (ANQP), which is a query-response protocol used by a mobile device to discover information including hotspot owner's domain name, roaming partners accessible via the hotspot, and EAP method used for authentication and IP address type availability [2, 3]. To support dynamic offloading in this paper, the roaming partners are updated dynamically according to the offloading decision of our algorithm.

Seamless mobility: There are several strategies to perform data offloading.

The simplest strategy is to use an application based switch, which simply moves a connection between cellular and Wi-Fi networks but does not worry about preserving the sessions across the switchover. It can cause a disruption to most applications and result in negative user experience, especially for VoIP, VPN applications, and video streaming applications [1].

3GPP Release 8 uses Dual Stack Mobile IP (DSMIP) to enable seamless handover between 3G and Wi-Fi. The solution does not require any support from Wi-Fi hotspots. The cellular radio access network supports a Home Agent (HA) that binds the new IP address of the node to the permanent IP. Since the IP address is preserved in this case, it provides a better user experience compared to application based switching [1]. Moreover, 3GPP release 10 uses DSMIPv6, which allows mapping multiple IP addresses to a single permanent IP address and supporting simultaneous use of Wi-Fi and 3G according to application QoS requirements (*e.g.*, keeping VoIP application on 3G and using Wi-Fi for bandwidth intensive applications like video streaming). [46] uses an implementation to quantify the efficiency of DSMIPv6 for managing handoffs between networks. It reports a 0.02 sec interruption and 3 lost packets when switching between IPv6 to IPv6 connection using 2 interfaces and 0.09 sec interruption and 17 lost packets while switching between IPv4 to IPv6 network. Further optimizations are possible to reduce the switching delay and packet losses. For example, [73] shows packet loss can be reduced to near zero with buffering on the mobile nodes.

Our implementation: We develop a prototype implementation on Linux

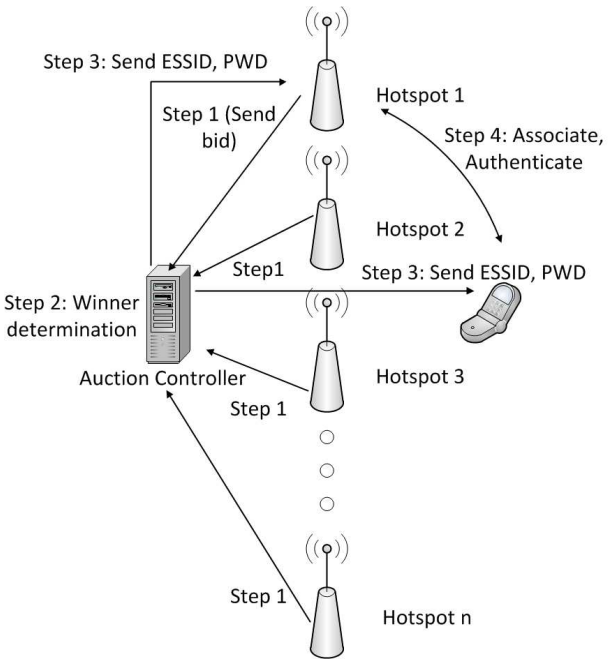


Figure 7.1: System Architecture.

machines using a NetGear WAG511 NIC to demonstrate the feasibility of our solution. Figure 7.1 shows our system architecture. Through a simple web interface, hotspot owners can submit their bids to the service provider machine, who controls the auction. Hotspots are configured using hostap [36]. Depending on who wins in the round, the service provider sends a message to the hotspot with the ssid and password it should use in the current round and also sends the ssid and password to the mobile client machine so that it can connect to the winning hotspot. This message is sent using TCP sockets. Authentication between mobile client and hotspot is done using WPA PSK through WPA Supplicant [88].

We collected performance statistics from the mobile client for billing

and keeping track of hotspot quality score. We measured the upload and download statistics on the wireless interface using the Collectl tool [19] periodically (*e.g.*, every 10 secs) and send back the data to service provider PC for bookkeeping. iDEAL allocation and pricing take 43 ms and 74 ms, respectively, which are both small.

We further measure the association and authentication time in our implementation. After getting the scan results, it takes 18 ms to associate, 103 ms to perform 4-way handshake (*i.e.*, defining individual keys for unicast transmission), and 3 ms to perform the group handshake (*i.e.*, defining keys for broadcast transmission). The authentication times can be further reduced (*e.g.*, using techniques in [41, 56]). Moreover scan times can be shortened by selective scanning on fewer channels as proposed in [66]. Therefore, we can achieve a very low overhead for handoff, making offloading feasible.

Chapter 8

Dynamic Spectrum Allocation via Double Auctions

8.1 Background

A double auction implements a double-sided market. The market consists of three types of entities: *buyers*, *sellers* and an *auctioneer*. Buyers submit *bids* which specify the item they are interested in and their maximum willingness to pay. Sellers submit *asks* which include the item they offer for sale and their asking price. The auctioneer evaluates the bids and asks and determine the winners and the items that are traded. The auctioneer also determines the amount to pay to the sellers and the amount to charge the buyers. All payments from buyers are paid to the auctioneer and the auctioneer pays the sellers. A buyer's utility is then the difference between his valuation of the item he wins and the amount he pays the auctioneer. Similarly for sellers.

Figure 8.1 shows a spectrum double auction. Here the sellers are spectrum resource owners (*e.g.*, cellular service providers or organizations that own spectrum resource). The buyers can be any entities that need more spectrum. To capture interference relationships among the buyers, measurement can be conducted by the buyers, sellers, auctioneer, or a third party to derive the con-

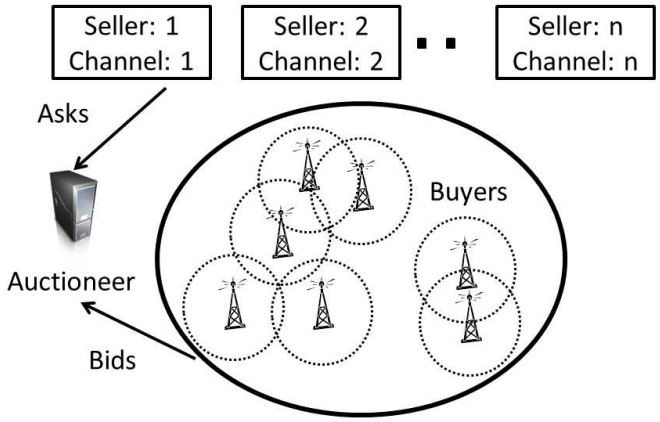


Figure 8.1: Double Auction: Spectrum Market

flict graph. We can leverage existing approaches to either actively or passively measure the conflict graph (*e.g.*, [62, 6, 75, 67, 94]).

After winning the auction, buyers then start using the new spectrum by switching some of their clients onto the new spectrum. Techniques in intra-cell handover (*e.g.*, [31, 32]) and spectrum virtualization (*e.g.*, [35, 78]) can be applied to efficiently and seamlessly switch users to the new spectrum.

Basic auction settings: We consider a double-sided spectrum market where the goods of interest are wireless channels (frequency bands) and the players are spectrum resource users, such as cellular service providers. We assume a seller offers one channel for sale and a buyer seeks to buy one channel as in previous works [95, 91]. Multi-unit double spectrum auction is much more challenging and we leave it for future work. A seller can sell to any buyer in the market. We relax this requirement in Section 9.4. The auction runs periodically to enable spectrum reallocation in a dynamic fashion. The frequency of the auction depends on the volatility of demands and the cost of running

an auction, which may include gathering inputs to the auction, computing the auction solution, disseminating auction outputs, and collecting payments.

Critical value is the lowest value that a buyer can bid and still win. *Monotonic allocation* means that a buyer that wins by bidding x will still win if it bids y where $y > x$. The definitions are similar for the seller side.

8.2 Challenges in Spectrum Double Auction

Challenges in supporting spectrum reuse: Spectrum auction is fundamentally different from a conventional auction in that each item in a conventional auction can only be used by one buyer whereas spectrum can be reused by multiple buyers as long as they do not interfere with each other.

Wireless interference dictates how spectrum can be reused, so it has significant impact on the design of spectrum auction. *Conflict Graph* [40] is commonly used to capture the interference relationship between the buyers, where each node in the conflict graph denotes a buyer and there is an edge between two nodes if the corresponding buyers interfere.

Wireless interference has the following major impacts on the spectrum auction. First, conventional auction design implicitly assumes all buyers compete against each other globally. In comparison, to support frequency reuse in spectrum auction, a buyer a only competes locally with the nearby buyers that interfere with a. Second, a buyer's competition is coupled with other buyers' competition depending on their interference. As an example, consider

the conflict graph: $a(7) - c(10) - b(5)$, where " $-$ " represents the interference between two buyers and the numbers are the bids. In this example, buyer a or buyer b individually does not outbid buyer c. However, since the buyers a and b do not conflict and can share the channel, letting them win together is a better choice than letting buyer c win alone. So a buyer's auction outcome not only depends on its competitors' bids, but also depends on its collaborators' bids, which further depends on its collaborators' competitors' bids and so on. This significantly complicates the auction design.

Challenges in designing a truthful double auction: As mentioned in Section 1, achieving truthfulness in double auction is very challenging. In particular, combining truthful seller side auction and truthful buyer side auction does not necessarily lead to truthful double auction. We illustrate this using the following example.

Consider the following buyer side design that tries to maximize efficiency on the buyer side by maximizing the sum of winning buyers' bids. This allocation criteria is monotonic. It is a truthful single sided auction when it uses critical value pricing. Suppose there are four buyers a, b, c and d, where a, b and c do not conflict with each other and they all conflict with d. The valuation of a, b and c is 3, and the valuation of d is 7. Assuming we have 1 channel to allocate, the winners are a, b and c because the sum of their valuation is 9, which is greater than 7. The critical value for each winner is 1, because one of the three nodes, say a, can win when it bids as low as 1 (as the sum of a, b, c is still as high as d's bid). So the total revenue is 3. However,

it is possible that on the seller side, when one channel is sold, the selling price is higher than 3, in which case budget balance is not satisfied and this trade cannot happen. So no buyer gets a channel, and everyone has 0 utility.

Now a lies and lowers its bid to 1. a, b and c still win but the critical values of b and c both become 3 and a's critical value is 1 in order for the sum of their critical values to still be as high as d's bid. The total revenue thus increases to 7. If this is higher than the price on the seller side, the channel will be traded and a receives a positive utility of $3 - 1 = 2$ (*i.e.*, the difference between its valuation and its payment). Thus the double auction is not truthful since a's utility increases when a lies about its valuation.

Weaknesses of existing solutions: There are a few previous works on this topic. The pioneering and most representative work is TRUST [95], and several variants have been proposed since then. Since they are similar to TRUST and share the same strengths and weaknesses, we focus on TRUST.

TRUST follows the classic McAfee's double auction design [54] to achieve truthfulness. McAfee's design is a single unit double auction. It first sorts sellers in an increasing order of their asking prices and sorts buyers in a decreasing order of their bids. Then it matches sellers to buyers from the two sorted list by matching the first seller to the first buyer, the second seller to the second buyer, and so on. It stops when all "profitable" matches are found, *i.e.*, matches in which the asking price is lower than the bid. Finally, it sacrifices the least profitable match. Sellers and buyers in all other profitable matches trade. For pricing, it uses the asking price and the bid in the sacrificed trade

as the price to pay to the winning sellers and the price to charge to winning buyers, respectively. Because the sacrificed match is also profitable (*i.e.*, its asking price is lower than the bid), budget balance is satisfied when they are used for pricing.

To use McAfee's design, TRUST divides buyers into multiple independent sets or groups, where the buyers in each independent set do not interfere with each other. For each group, TRUST computes the group bid as the lowest bid in the group multiplied by the group size. Then McAfee is applied by treating each group as one virtual buyer. As in McAfee, all winning groups pay the group bid of the group in the sacrificed match. Within the group all members share the price equally. TRUST enables spectrum reuse by grouping buyers and achieves truthfulness by applying McAfee's design. However it has the following weaknesses.

- Low efficiency: The independent sets are constructed in a random fashion. It is possible that a high bidding buyer loses just because it is put into a bad independent set. As a result, the final winning groups can be suboptimal.
- Low revenue: The group bid only depends on the lowest bid in the group and all higher bids are ignored. So the group payment is significantly limited by the lowest bid, which can be much lower than other bids in the group.
- Unfairness: TRUST uses uniform price for the buyers that are assigned the same channel. But buyers assigned the same channel may have differ-

ent interference (competition) patterns. The uniform pricing forces some winners to pay for other winners' competition.

We illustrate the above weaknesses using the following example. The conflict graph and bids are: $a(60) - b(2) - c(30) - d(100)$. Assume two sellers each offering one channel: channels 1 and 2, respectively. Let (a, c) and (b, d) be the independent sets generated by TRUST. The group bid of (a, c) is $30 \times 2 = 60$, while the group bid of (b, d) is $2 \times 2 = 4$. Thus (a, c) is matched with seller 1 and (b, d) is matched with seller 2. (b, d) is the last profitable match, so only (a, c) wins. a and c together pay the group bid of (b, d), which is 4. So a and c each pays 2.

In this example, because d is in the same group as b, his bid is ignored although it is the highest. a and d winning together gives the highest efficiency in this example. However, TRUST fails to consider that possibility due to the group bid computation. TRUST also yields low revenue in this example since it only uses the lowest bid, while the winning bidders can potentially pay much more. TRUST is also unfair to d in this example, because he cannot win regardless of how high he bids.

There have been several works on improving TRUST [89, 91]. Although these works alleviate certain problems of TRUST, they are still similar in spirit and still use random independent sets and enforce uniform price for buyers assigned the same channel. So they still suffer from low efficiency and poor fairness.

Chapter 9

Our Solution: DA^2

9.1 Overview

Design strategy: We develop a novel double auction for dynamic spectrum allocation. It consists of the following three parts: (i) seller side auction design, assuming N channels sell (ii) buyer side auction design, assuming N channels sell, and (iii) a procedure to determine the number of channels N to sell to satisfy budget balance.

To the best of our knowledge, our solution represents the first double auction design for spectrum allocation that explicitly decouples the buyer side and seller side auction design while achieving (i) truthfulness, (ii) individual rationality, and (iii) budget balance. Previously, seller side and buyer side auctions have to be designed jointly in order to satisfy all three properties and (*e.g.*, McAFee's principle as used in TRUST all its variants).

Decoupling seller side and buyer side design is crucial for dynamic spectrum allocation for two reasons. First, using separate auction designs for the buyers and sellers, we no longer require groups of buyers that share the same channel to be formed in advance to match with sellers one by one. Instead we can make more informed decisions based on the bids to select a stronger set

of winners and improve efficiency, revenue, and utilization. Second, it enables flexible combination of different buyer/seller side designs. This is especially beneficial in our context since the two sides have rather different properties. For example, it is commonly assumed that a seller can sell to any buyer in the auction. Competition between sellers is similar to traditional auctions, whereas the buyers' competition is much more complicated due to complicated wireless interference.

Proof strategy: The key to our ability to design two sides separately is a new proof strategy for establishing truthfulness in a (decoupled) double auction. Formally we have the following theorem:

Theorem 8. *A double auction for dynamic spectrum allocation is truthful if the following two conditions hold: (i) both seller side and buyer side auctions are truthful when the number of channels that are sold, denoted as N, is fixed, and (ii) no seller or buyer can improve its own utility by unilaterally modifying its own bid and causing N to change.*

The correctness of this theorem is easy to see: When a bidder lies but does not change N, he cannot gain because both sides are truthful when N is fixed. When a bidder lies and changes N, he can not gain either because no seller or buyer can improve its own utility by unilaterally modifying its own bid and causing N to change. Thus a bidder never gains by lying and the auction is truthful.

We use the following theorem from [47] to assist our design:

Theorem 9. *If every losing bidder (*i.e.*, unallocated) in an auction pays nothing, it is truthful if and only if its allocation algorithm is monotonic and it uses critical value as payment.*

Design overview: Below is an overview of our design. The buyer side auction is complicated since buyers' competition is determined by the complex conflict graph and buyers without interference can share a channel. This is fundamentally different from traditional auctions. Therefore this paper focuses on the buyer side auction design.

The seller side auction is a standard auction, since a seller can sell to any buyer in the auction. Therefore we can apply uniform pricing or VCG for single-unit auction. We prove it satisfies the properties specified in Theorem 8.

To determine the number of channels N, we start by setting N as the total number of channels that sellers collectively have. We run seller and buyer side auctions separately, and then check if budget balance is satisfied (*i.e.*, the payment collected from the buyers is no less than the payment to be paid to the sellers). Note the payment is determined by our seller side and buyer side auction design described below and not the sum of winning bids/asks. If budget balance is already satisfied, we terminate. Otherwise, we decrease N by 1, and run the auction again. By reducing N, it requires buyers to bid even higher to win and sellers to ask even lower, thus reducing the gap between revenue and payment. We stop when the budget balance is satisfied and N channels are then traded. This procedure always terminates because budget

balance is satisfied when N drops to 0. Thus we guarantee budget balance, which is formally stated in the following theorem.

Theorem 10. *Our design satisfies budget balance.*

Note that McAfee's design is a special case of our framework, where both the buyer side design and the seller side design use uniform pricing. This shows that our framework is general and under our framework it is possible to design a double auction that satisfies all economic properties. Below we describe our design in detail.

9.2 Seller side design

The seller side design is a standard auction since there is no need to consider interference among sellers. Thus we simply use the traditional uniform price design. Assuming N channels are sold, the N sellers with the lowest asking prices win, and they each get paid at the $N+1$-th seller's asking price. This uniform price design is known to be truthful in a single unit auction when N is fixed. Moreover, since sellers are paid higher than their asking price, individual rationality is satisfied. We further prove theorem 11 to ensure this design is truthful when applied to a double auction. See the Appendix for proof.

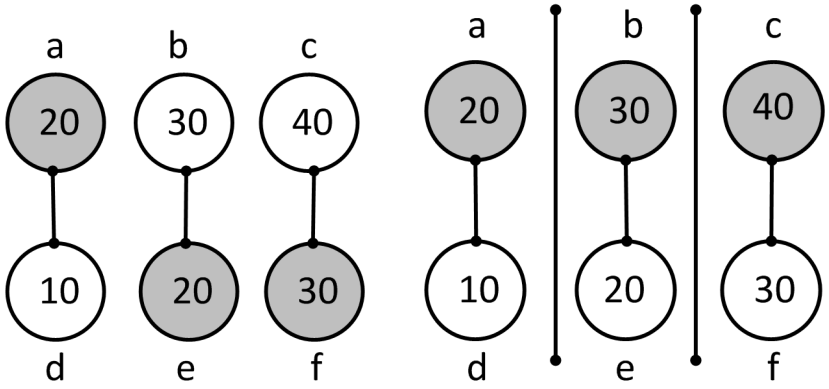
Theorem 11. *This seller side design, when applied to double auctions, does not allow a seller to unilaterally manipulate N (*i.e.*, the number of channels that can be sold) and gain.*

9.3 Buyer side design

Overview: To ensure fairness and achieve high revenue, it is important for the buyer side auction to explicitly take into account the fact that different buyers face different levels of competition depending on their locations and interference pattern. Existing approaches do not account for such individuality and apply uniform pricing, where all buyers that share the same channel pay the same amount. This is not only unfair, but also reduces revenue since a buyer's payment is limited by the lowest bids among the buyers that share the channel.

Our key idea is thus divide-and-conquer. Specifically, we first partition the conflict graph into subgraphs. Independent sets are then constructed within each subgraph and pricing is computed independently in each subgraph. We then design a merge strategy to combine the winners from different subgraphs while preserving truthfulness. Note that when computing the allocation within each individual subgraph, we ignore the inter-subgraph conflict edges. During the merge procedure, we need to add back the inter-subgraph conflict edges. As a result, some winners from individual subgraphs may have to be removed due to the inter-subgraph conflict edges. The challenge is how to do so without compromising truthfulness.

Benefit of graph partitioning: Note that while our partition idea is inspired by clustering structure in realistic conflict graphs, it also benefits general graphs without explicit clustering structure. Before going to the details of our



(a) Without partition (b) With partition

Figure 9.1: Benefit of graph partition

design, we first illustrate the benefits using an example shown in Figure 9.1. It shows a conflict graph with 6 nodes a to f. We ignore the seller side in this example and assume we want to allocate 1 channel. Partitioning has two advantages: (i) better independent set construction, and (ii) higher revenue due to the removal of the uniform pricing constraint.

We first illustrate the benefit (i). Assume the groups constructed by TRUST are $G1$: (a, e, f) and $G2$: (b, c, d). The group bids of $G1$ and $G2$ are $20 \times 3 = 60$ and $10 \times 3 = 30$, respectively, which are the lowest bid times the group size. So in TRUST $G1$ wins and pays the first losing group's bid, which is 30. The efficiency on the buyer side is the sum of all winning buyers' bids, which is $20 + 20 + 30 = 70$, and the revenue is 30. Now if we treat each subgraph independently as shown in Figure 9.1(b): a, b and c win in their subgraphs and they each need to pay the first losing buyer's bid, which are 10, 20, and 30, respectively. Then we get the efficiency of $20 + 30 + 40 = 90$ and revenue of $10 + 20 + 30 = 60$, both of which are much higher than TRUST.

To demonstrate the benefit (ii), we consider TRUST happens to construct the same groups as ours, namely $G1'$: (a, b, c) and $G2'$: (d, e, f). The group bids of $G1'$ and $G2'$ are $20 \times 3 = 60$ and $10 \times 3 = 30$, respectively. So $G1'$ wins, and pays the first losing group's group bid, which is 30 and still lower than the revenue in our scheme, because the revenue of TRUST is limited by the uniform pricing (*i.e.*, the lowest bid times the number of buyers). The difference in revenue may have even bigger impact when we take the seller side into consideration, because it affects the number of channels that can be sold to maintain budget balance. Fewer channels can be sold if revenue is too low, which in turn further reduces efficiency and revenue.

Essentially partitioning allows us to decouple nodes with no or weak interference into different subgraphs so that we can improve the independent set construction and avoid unnecessary coupling in different buyers' pricing even though they interfere with very different sets of nodes.

Design questions: Several important questions should be addressed in order to realize the benefits of graph partition:

- How to partition the graph so that we can retain important interference relationships within a subgraph and decouple nodes with weak and no interference into different subgraphs?
- How to compute auction results within each partition?
- How to merge the auction results from different partition to achieve truthfulness and budget-balance?

Below we answer each question in turn.

9.3.1 Graph partitioning

Given a graph A, a graph partitioning algorithm strives to find a partitioning A_1, A_2, ..., A_k that minimizes a certain objective. For the purpose of dynamic spectrum allocation through double auctions, a good graph partitioning algorithm should balance two key requirements: (i) the number of inter-subgraph edges should be small, and (ii) different subgraphs should be similar in size. If a subgraph is too small (*e.g.*, having only one buyer), a small number of channels can satisfy all its buyers, leaving no losing buyers and no revenue from the subgraph. On the other hand, a too big subgraph may lead to poor independent set construction and poor performance due to the uniform pricing in each group.

Two common alternative objectives have been proposed in the literature – RatioCut and Normalized cut (NCut). The former normalizes the weights of the edges on the cut by the number of vertices in each partition, namely it minimizes $\sum_{i=1}^{k} \frac{W(A_i,\overline{A_i})}{|A_i|}$, where $W(A_i, \overline{A_i})$ represents the total weight of all edges between A_i and the remaining nodes (*i.e.*, $\overline{A_i}$) and $|A_i|$ denotes the number of vertices in A_i. The latter normalizes the weights of the edges on the cut by the sum of node degrees in each partition, namely it minimizes $\sum_{i=1}^{k} \frac{W(A_i,\overline{A_i})}{vol(A_i)}$, where $vol(A_i)$ denotes the sum of degrees of all nodes in the partition A_i. It is easy to see $\frac{1}{|A_i|}$ and $\frac{1}{vol(A_i)}$ are minimized when either the number of vertices or the sum of node degrees within each partition is the

same. This captures our goal of finding balanced cuts while minimizing the weights of edges on the cut.

Minimizing either RatioCut or NCut is a NP-hard problem. Spectral clustering is a well-known effective scheme to find approximate solutions to this NP-hard problem (see [53] for a nice tutorial). There are many variants of spectral clustering [53, 55, 60]. In this paper, we use the Meila-Shi algorithm [55], which is the recommended algorithm in [53] due to its excellent performance and solid mathematical foundation. Let W be the adjacency matrix, with weight w_{ij} on its i-th row and j-th column. Let D be the degree matrix, which is a diagonal matrix with the node degree $d_i = \sum_j w_{ij}$ on the diagonal. The Meila-Shi algorithm takes the eigenvectors corresponding to the k smallest eigenvalues of the normalized graph Laplacian matrix $L_{\text{rw}} = I - D^{-1}W$ (where I is the identity matrix) and then invokes another algorithm (*e.g.*, k-means clustering [51]) to cluster points by their respective k components in these eigenvectors.

To automatically determine the number of clusters to create (*i.e.*, k), we follow the suggestion of [53] and apply the eigengap heuristic. Specifically, let $\lambda_1 \leq \lambda_2 \leq \cdots$ be the eigenvalues of the normalized graph Laplacian matrix L_{rw} (sorted in an ascending order). The eigengap heuristic computes all the eigengaps (*i.e.*, difference between two successive eigenvalues) and chooses the number of clusters k such that $(\lambda_{k+1} - \lambda_k)$ is the largest eigengap.

Note that when the conflict graph is disconnected, we first divide it into multiple connected components since nodes in different connected com-

ponents have no competition at all. Then we apply spectral clustering to each connected component to further partition the connected component.

9.3.2 Allocation within a subgraph

When a good partitioning is found, we first compute the allocation in each subgraph independently. For that we can apply existing algorithms, such as TRUST. To further improve the performance, in our implementation we apply the allocation algorithm proposed in TDSA [91]. It is similar to TRUST but it defines a new group bid and allows a subset of a group to win while the rest lose. Consider a group k and assume its members are sorted in a decreasing order of their bids. Denoting members as 1 to m, the group bid is defined as $\max\{b_i \times i | i = 1...m\}$.

The intuition of this group bid is that it quantifies the maximum potential payment of a group, if we allow a subset of this group to win. For each subset, the maximum potential payment is the lowest bid in the subset times the size of the subset. Thus this group bid finds the highest potential payment by enumerating all possible sizes of the subset. For example, if a group contains bids (1, 3, 5), its potential payment could be either letting 5 win alone, which corresponds to payment of $5 \times 1 = 5$; or letting 3 and 5 win together, which yields a payment of $3 \times 2 = 6$; or letting all of them win, which yields a payment of $1 \times 3 = 3$. The maximum payment achieved is 6 in this example.

To allocate N channels, the groups with the top N group bids win.

A winning group is then charged the first losing group's group bid and all members in the group share the price equally. If a member i cannot afford its fair share, *i.e.*, $b_i * i$ is smaller than the first losing group bid, i does not win and the price is shared among the remaining group members. Since the winning group has a higher group bid than the losing group, there always exists a subset of the group such that they can all afford their fair share. We prove that this procedure finds the critical value of a buyer under our partition and merge framework in Theorem 13 in the Appendix.

We also make a temporary assignment by sorting the channels in an increasing order of their asking prices and assign the first channel to the first group and second channel to the second group, and so on. However, this assignment is subject to change in the merge procedure below.

9.3.3 Merge strategy

Next we describe how to merge allocation results from different subgraphs. We merge results from two subgraphs at a time. The input of merge is the winners selected from the two subgraphs, including their channel assignment. The purpose of the merge is to find one way to reorder the channel assignment in one subgraph such that winners on the cut do not have conflict. In case such reordering does not exist, certain nodes on the cut may be dropped. We propose the following bid-independent merge strategy to select a node to drop. We prove this merge strategy preserves truthfulness.

Figure 9.2 gives the pseudo-code of the merge procedure. In lines 1-4,

```
1   for both A and B
2       Find all winning buyers on the cut
3       Merge buyers sharing the same channel into a virtual buyer
4       Put all winning buyers into Candidate
5   for each i in Candidate
6       Let degree_i be number of i's edges on the cut
7   flag = a feasible reordering function reorder(x) can be found,
    where x is a channel currently used in B.
    Note reoder(x) is only defined for channels that need to be replaced.
8   while flag is false
9       Sort Candidate in a decreasing order of degree_i
10      Drop Candidate[0]
11      Update effective degrees
12      flag = a feasible reordering function reorder(x) can be found
13  for each winning buyer i in B
14      if reorder(assign_B(i)) is defined
15          assign_B(i) = reorder(assign_B(i))
16  assign_AB = assign_A ∪ assign_B
17  return assign_AB
```

Figure 9.2: Pseudo code for bid-independent merge.

we preprocess the buyers on the cut by combining the buyers that share the same channel on each side into one node to ensure buyers sharing the same channel in one subgraph is always assigned the same channel no matter how the assignment is reordered. Lines 5-6 compute the *effective degree* for each buyer i, which denotes the number of i's edges on the cut. This information is used in lines 7-12 to determine which buyer to drop when a feasible reordering does not exist. We drop buyers in a decreasing order of their effective degrees in order to minimize efficiency and revenue loss. Every time a buyer is dropped, the effective degrees of the remaining nodes are updated. We iterate until a feasible reordering is found, and then use it to derive a joint assignment for the union of A and B. Intuitively, our partition and merge framework preserves truthfulness because both operations by themselves are bid-independent and there is no incentive to lie. It is still possible to lie to change the allocation in subgraphs, which changes the input of the merge operation, but a buyer

cannot gain this way because of the use of critical value pricing. We formally prove truthfulness in Theorem 12.

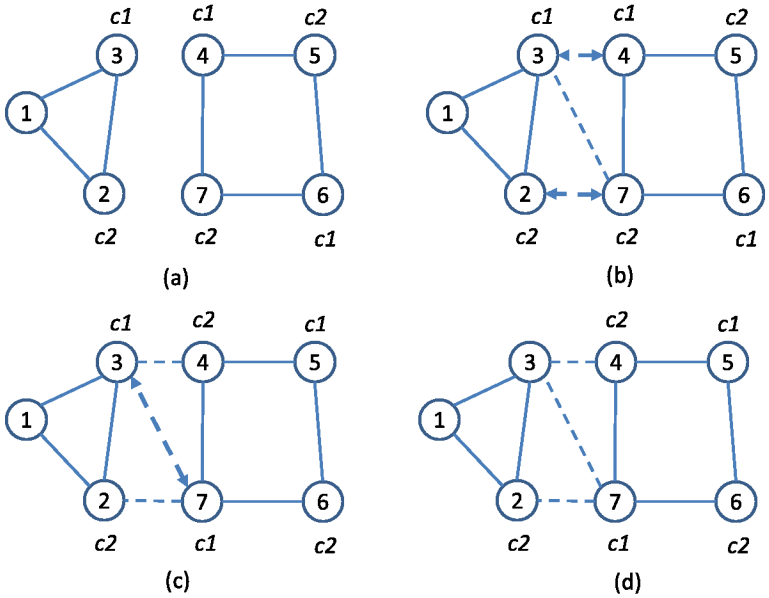


Figure 9.3: A simple example of the merge procedure

Figure 9.3 shows an example of merging two subgraphs. Let A and B denote the left and right subgraphs, respectively, and $c1$ and $c2$ denote the channels to be allocated. The labels next to nodes represent the channel assignments to the nodes. Figure 9.3(a) shows the assignment to the winners in each subgraph. If in one subgraph, some winning buyers on the cut share the same channel, they should be merged to form a virtual buyer before we proceed. There's no such buyers in this simple example so we go to the next step by adding back the removed edges. In Figure 9.3(b), the three previously removed edges are added back and there is conflict between nodes 3 and 4 and conflict between nodes 2 and 7. In Figure 9.3(c), the algorithm tries to reorder

the assignment in B by swapping $c1$ and $c2$ but there is still conflict between nodes 3 and 7. So in Figure 9.3(d) node 3, which is the node on the cut with highest degree, is dropped to resolve the conflict.

Theorem 12. *DA^2 is truthful.*

Proof. To prove this theorem, we first prove Theorems 13 and 14 in the Appendix, and then apply Theorems 8 and 11. □

9.4 Incorporating seller locality

Motivation: So far our solution assumes a seller can sell to all buyers. In reality, a seller may own channels in a limited area and cannot sell to all buyers. This problem is considered in [84], but it requires each buyer-seller pair to be budget balanced, which is a very strong requirement and can significantly limit spectrum reuse because the new buyer may not be able to afford the entire cost of a channel. An important advantage of a spectrum double auction is that we can leverage payment from multiple buyers that reuse the same channel to pay to a seller.

Challenges: A naive approach is to divide the global area into regions where each region is covered by a few sellers and in each region sellers can sell to all buyers in that region. Then our allocation can be applied to each region independently, assuming each buyer only belongs to one region. However, treating each region as an individual auction can prevent feasible trades and limit spectrum reuse because the revenue in a single region may not be enough

to satisfy the sellers' price (*i.e.*, budget balance is violated within one region), while the revenue from multiple regions together may be sufficient to pay for the sellers (*i.e.*, budget balance is satisfied globally). Moreover, uniform pricing for sellers may not be truthful since only the sellers whose coverage overlap compete against each other. Different sellers face different competitions.

To address these challenges, we change the seller side design of the auction to use *Cross-region budget balance* and *Non-uniform seller side pricing*. In *Cross-region budget balance*, we allow a region that has sufficient revenue to help other regions. We illustrate the idea using a simple example. Consider a region with three sellers 1, 2 and 3, and sellers 2 and 3 are already budget balanced in other regions (*i.e.*, sellers 2 and 3 get enough payment from the buyers in other regions). Then we do not need to consider budget balance requirement for sellers 2 and 3 in this region, and this region is budget balanced as long as the revenue is higher than the payment to seller 1. In this way, we can allow more channels to be sold and achieve higher efficiency and spectrum reuse. In *Non-uniform seller side pricing*, we compute the critical value of winning sellers under our new allocation algorithm. We describe them in detail below.

Cross-region budget balance: To generalize the intuition from the previous simple example, we design the scheme in Figure 9.4. At a high level, it finds regions that are budget balanced and use them to lower the budget requirement for the other regions that overlap with the budget-balanced regions. As shown in Figure 9.4, lines 1-5 initialize all regions and all sellers to be not

```
1   for each seller i
2       SellerBalanced_i = 0
3       Lose_i = 0
4   for each region m
5       RegionBalanced_m = 0
6   while not all regions are budget balanced
7       find seller j with the highest ask price a_j
        and SellerBalanced_j = 0
8       Drop j by setting Lose_j = 1
9       for each region m where RegionBalanced_m = 0
10          Compute allocation assuming all remaining sellers in m win
11          Compute buyer pricing and the total revenue R
12          Compute the total selling price:
            S = sum_{i in m, SellerBalanced_i = 0} a_j
13          if R >= S
14              RegionBalanced_m = 1
15              for each seller i in m
16                  SellerBalanced_i=1
17  Allocation completes and all sellers with SellerBalanced_i = 1 win.
```

Figure 9.4: Pseudo code for allocation with cross-region budget balance.

budget balanced, and all sellers are initially included in the auction. In lines 6-17, we drop the highest asking seller that is still not budget balanced until all regions are budget balanced. Because the selling price is determined by the losing sellers' prices, dropping a seller lowers the selling price and makes it easier to satisfy budget balance. Every time a seller is dropped, in lines 9-13 we compute allocation in each region that is still not budget balanced and check if budget balance is satisfied with the new allocation, assuming any seller that is not already budget balanced will be paid at the last dropped seller's asking price. For sellers that are already budget balanced, we consider the payment as 0. In lines 14-17, if a region becomes budget balanced in this round, we label all its sellers as budget balanced and we also label this region as budget balanced. Finally, all the sellers that are budget balanced will win.

With cross-region budget balance, the point where budget balance is satisfied is no longer characterized by a single N. Instead it is a vector of

numbers N_i, where N_i the first losing seller's position in region i.

Non-uniform seller side pricing: It is easy to see that the above allocation is monotonic for sellers. Then in order to achieve truthfulness, we pay the winning sellers their critical values. If a seller becomes budget balanced after seller j is dropped, its critical value is the j's asking price. Different sellers' critical value may differ, which is desirable, since it captures their different competition patterns. We prove that is indeed the critical value in Theorem 15 in the Appendix.

According to Theorem 9, we conclude that the seller side is truthful if the budget balance point does not change. To show that the double auction is truthful we require that no seller can change the budget balance point by lying and gain. The proof is similar to the proof of Theorem 11 and omitted.

9.5 Practical Issues

Next we discuss practical issues involved in applying DA^2.

Reputation score: The auctioneer maintains a reputation score r_i ($0 < r_i \leq 1$) for both buyers and sellers to reflect their quality. For a seller the score can be based on his channel quality and for a buyer it can be based on whether this buyer uses wireless resource carefully without causing extra interference to other buyers. The higher the score, the better the reputation. When computing the allocation, we divide every seller's asking price by its reputation score so that it is harder for a seller with bad reputation to win.

We then multiply a seller's reputation score to its critical value to compute its final payment. Since the critical value is computed based on the reputation weighted asking prices and is greater than a winner's asking price divided by its reputation, individual rationality is achieved. Similarly, for buyers we can multiply their bids with their reputations when computing the allocation, and then divide its critical value by its reputation score to get the real price they need to pay to ensure individual rationality. Truthfulness is also preserved as the reputation scores do not depend on bids. Moreover, budget balance is still satisfied because we determine N based on the the real selling price and payment (after multiplying/dividing the reputation scores).

Leveraging prior knowledge: The independent set construction is critical to the performance. Our solution mitigates the randomness but the construction can be further improved if the auctioneer has prior knowledge, *e.g.*, distributions of buyers' valuations. Specifically, we can maximize the expected valuations in winning groups, which directly relates to auction efficiency by formulating the construction as a *maximum weight independent set* problem (MWIS). The expected valuation of an independent set is simply the sum of expected valuations of all the members. The MWIS problem on conflict graphs can be approximated in polynomial time, *e.g.*, [61]. When the error of prior knowledge with respect to the actual bids is up to 10%, the efficiency improves by 16% over our current scheme on average; and when the error increases to up to 50%, the average improvement becomes 6%.

Chapter 10

DA^2 Evaluation

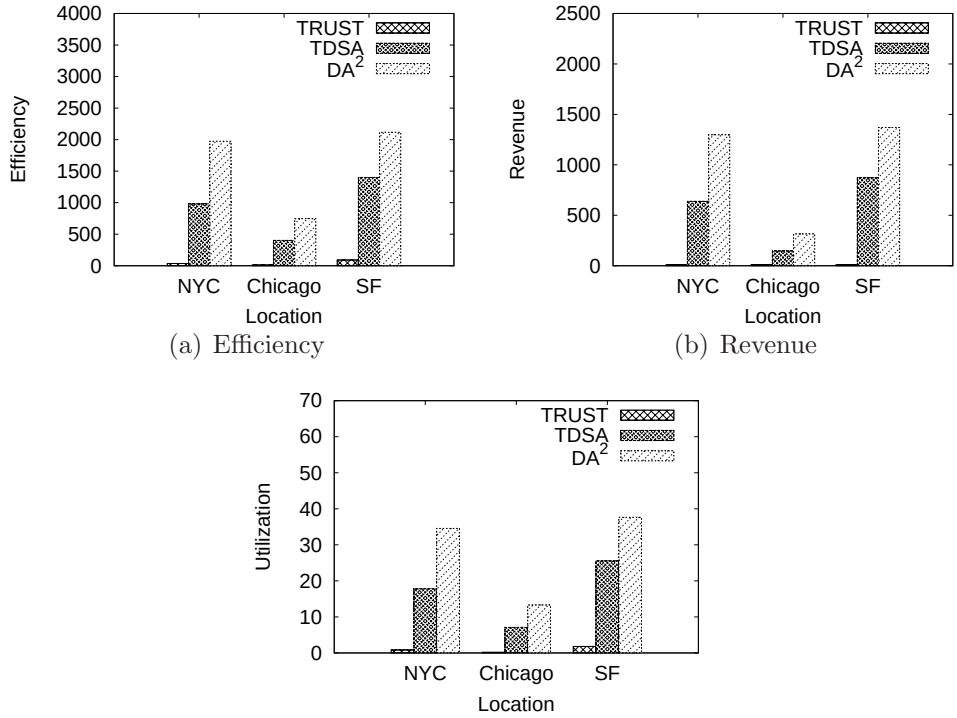
Simulation setup: In order to experiment with realistic conflict graphs, we use the location data of cell towers from a large US service provider. We consider there are buyers at each tower/location looking for spectrum resources at that location and that they do not collude. We construct the conflict graph for three cities: New York City (NYC), San Francisco (SF) and Chicago. In each city, we pick a grid (approximately 5km by 5km) encompassing the downtown area and use all the cell towers in the grid to generate the conflict graphs. We consider that two nodes conflict if the inter-node distance is smaller than 500m, which is considered a typical cell range. We also vary the range to see how the network density impacts the performance. We only present results with conflict graphs generated based on real locations but we also experimented with random conflict graphs and observed similar benefits.

We use 5 sellers by default, and also vary the number from 3 to 7 to see the impact. For the sellers' asks and buyers' bids, the absolute values do not matter and only their ratio matters. We generate the buyers' bids drawn from a uniform distribution between 0 to 100. We also use a uniform distribution to generate the asking prices. Since each grid can cover at most 25 buyers and

we assume buyers and sellers value the sepctrum resource similarly such that the price from the two sides compare fairly, we generate the asking prices to be between 0 to 2500 such that the mean is 25 times of the mean of a buyer's bid. We also scale the sellers' asking prices to see its impact in our evaluation.

We compare our scheme with TRUST and TDSA [91] in terms of the following three metrics: (i) *Efficiency:* This is widely used to quantify auction performance. It is defined as the difference between the sum of all winning buyers' bids and the sum of all winning sellers' asks. (ii) Revenue: It is defined as the total payment from all winning buyers. Since the payment uses critical value, which is different from the bids, revenue is also different from the efficiency on the buyer side. A higher revenue gives a stronger incentive for sellers to participate. (iii) Utilization: This is defined as the number of winning buyers. This is a unique metric in spectrum auctions because the spectrum resource is precious but reusable, and a higher utilization means more winners can utilize the spectrum at the same time, which is preferred. For every setting, we run 20 times with different random asking prices and bids, and report the average.

Performance at different locations: We first compare the performance in all three cities. We find that the performance can differ significantly in different cities. Specifically, SF gives the highest values in all three metrics while Chicago gives the lowest and NYC lies in between. The performance difference is primarily due to difference in the numbers of buyers. SF has 16% more buyers than NYC, while the number of buyers in Chicago is only around

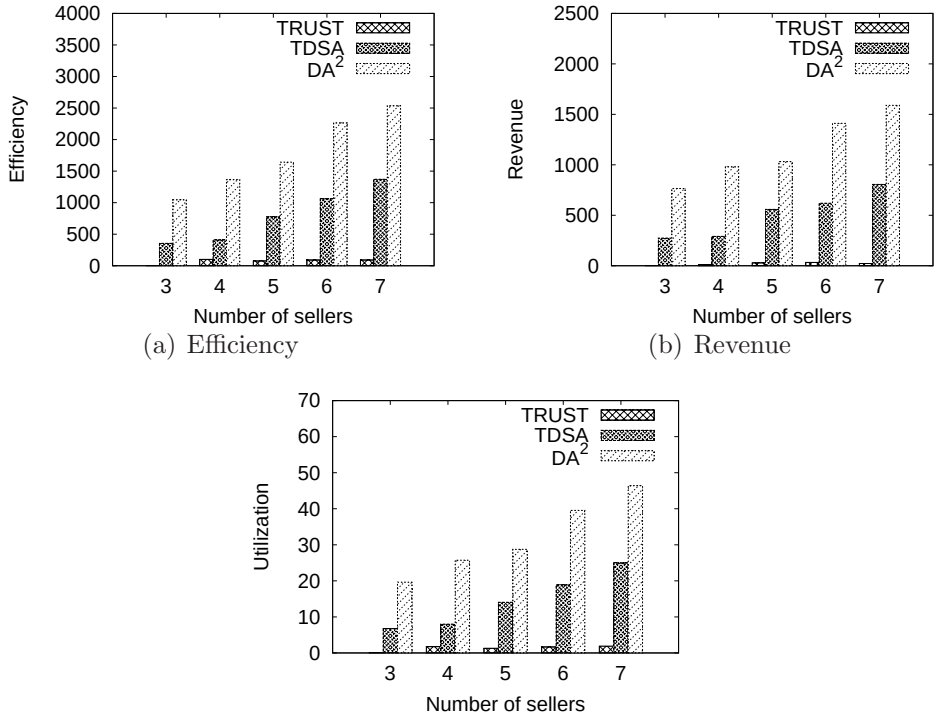


(a) Efficiency (b) Revenue

(c) Utilization

Figure 10.1: Performance at different locations

half of that in NYC. In general, more buyers tend to generate higher revenue, thus more channels can be traded. Among all the schemes, DA^2 achieves the best performance in all three metrics and for all three cities. It improves efficiency to 22x to 62x of that of TRUST, revenue 27x to 126x, utilization 42x to 65x. TRUST does not perform as well because it is limited by the uniform pricing and the lowest bid in each group. As a result, its revenue is low and only few channels can be sold. TDSA performs better than TRUST because it searches for the best subset of a group to win and is thus more robust to the lowest bid in a group. DA^2 still outperforms TDSA by 51% to 101% in efficiency, 57% to 115% in revenue, and 47% to 93% in utilization. This is



(a) Efficiency (b) Revenue

(c) Utilization

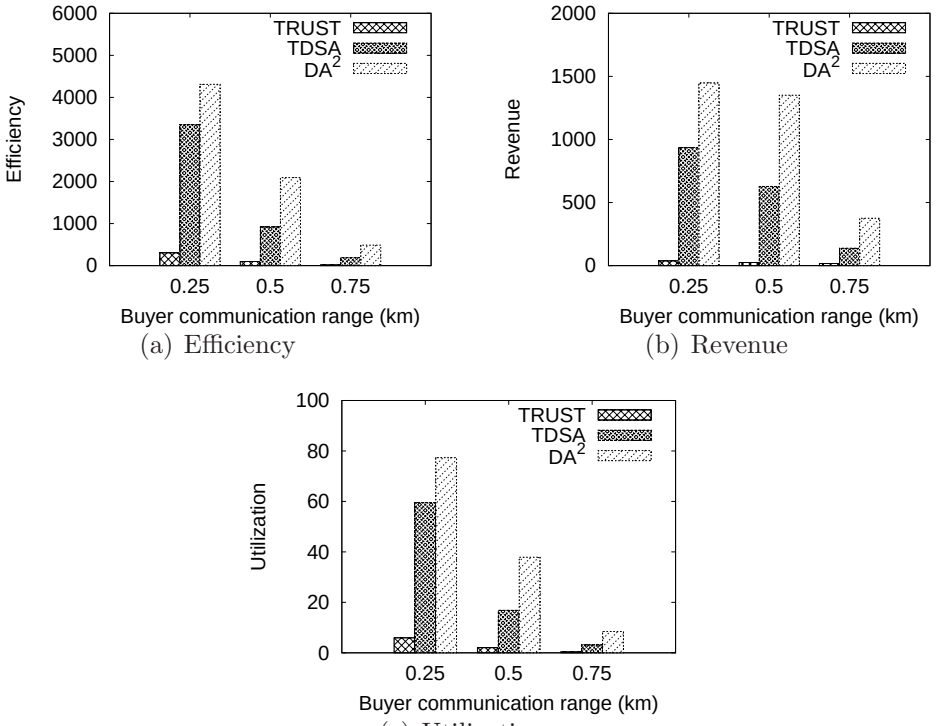
Figure 10.2: Impact of the number of sellers

because DA^2 (i) decouples pricing in different subgraphs to better capture the competition, and (ii) combines the top groups from different subgraphs to reduce randomness and find a set of winners with higher valuations.

In the remaining evaluation, we use the NYC conflict graph as default and study the impact of other parameters.

Impact of the number of sellers: We now vary the number of sellers from 3 to 7. As shown in Figure 10.2, with an increasing number of sellers, more channels become available and the price reduces due to increased seller-side competition. As a result, more channels can be traded and all performance metrics improve. DA^2 consistently out-performs the existing approaches. Its

improvement is highest when there are only 3 sellers, in which case they achieve as much as 3x times the performance of TDSA in all three metrics. TRUST does not sell a single channel in the 20 runs with 3 sellers because its revenue is low and it is more challenging to sell a channel while ensuring budget balance when the number of sellers is small. When the number of sellers increases to 7, DA^2 out-performs TRUST by 27x in efficiency, 71x in revenue, and 23x in utilization; and out-performs TDSA by 85% in efficiency, 97% in revenue, and 86% in utilization.

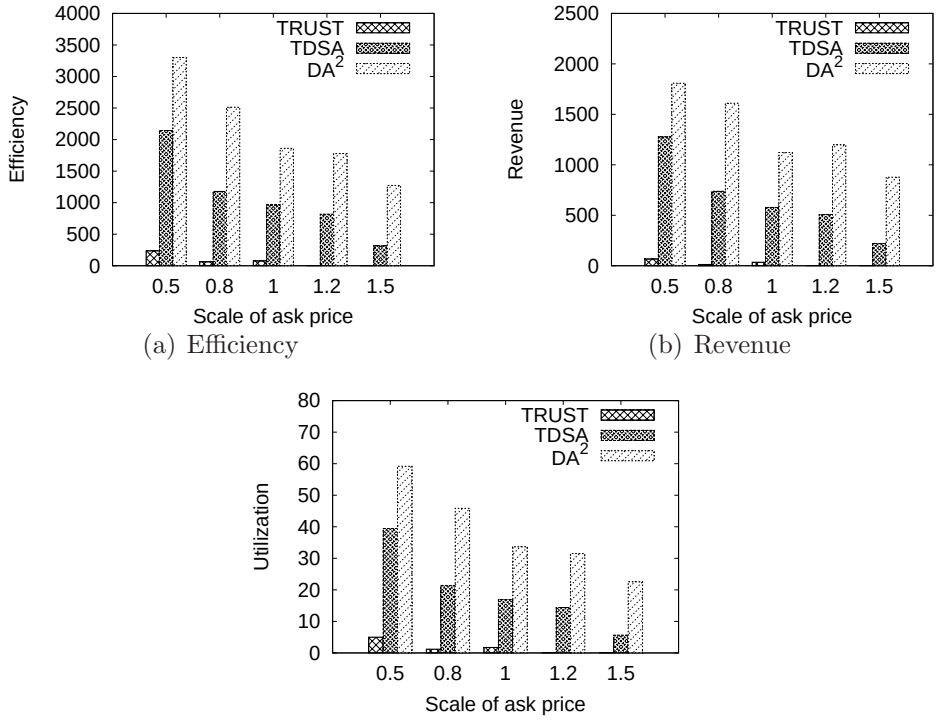


(a) Efficiency (b) Revenue

(c) Utilization

Figure 10.3: Impact of network density

Impact of network density: Next we vary the network density by changing the buyer communication range from 250m to 750m. A longer range indicates



(a) Efficiency (b) Revenue

(c) Utilization

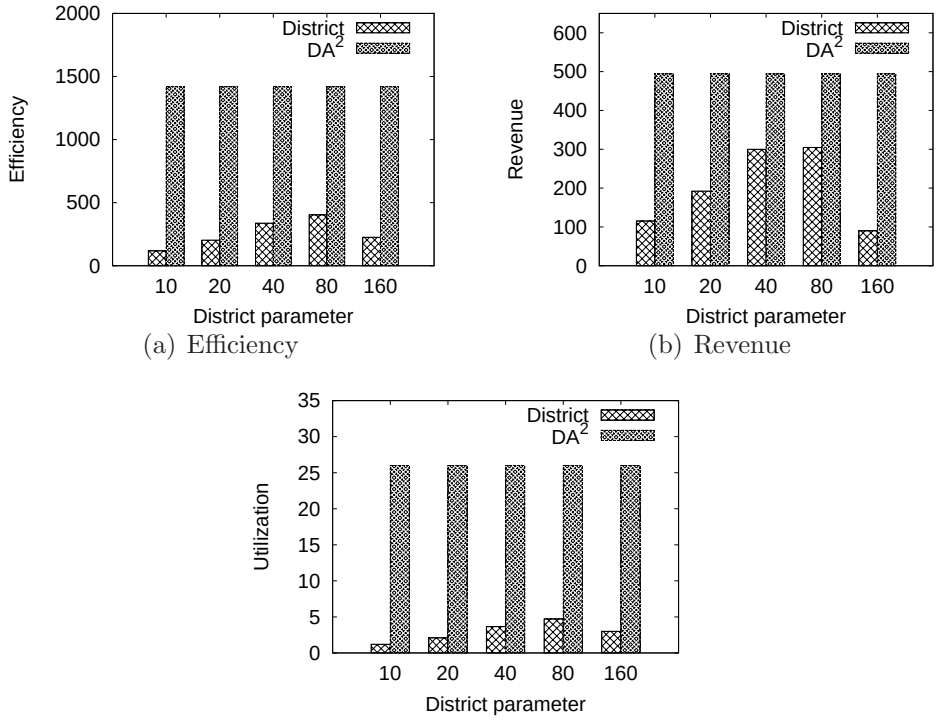
Figure 10.4: Impact of bid distribution

more competition among buyers and fewer buyers can reuse a channel. As shown in Figure 10.3, the benefit of our scheme increases with the range since it is harder to sell a channel and a good auction design becomes even more important. For example, when the range is 250m, our scheme out-performs TDSA by 29% in efficiency, 55% in revenue and 30% in utilization. The corresponding numbers for a range of 750m are 152%, 172% and 173%. The performance trend in TRUST is less clear because TRUST heavily depends on the random independent set construction. However, under all three ranges, DA^2 achieves 14x the efficiency, 22x the revenue, and 13x the utilization of TRUST.

Impact of bid distribution: To understand the impact of bid distribution, we scale all the sellers' asking prices by a factor between 0.5 to 1.5 after the initial ask is drawn from the uniform distribution with values between 0 and 2500. This changes the ratios between sellers' valuation and buyers' valuation. A lower asking price means budget balance is easier to achieve. A higher asking price requires a higher revenue from the buyer side in order to sell the channels, and is more challenging for an auction scheme. Figure 10.4 clearly shows that trend. In all cases, DA^2 outperforms the existing schemes. The benefit of DA^2 increases with the asking price. For example, when we scale the asking price to 1.5 times, our approach out-performs TDSA by 3 times in all three metrics. The improvement over TRUST is even larger, as TRUST does not sell a channel in this case and all its metrics are 0.

Seller locality: To study localized sellers, we compare DA^2 with the District algorithm described in [84], which is specifically designed to incorporate seller locality. For fair comparison, we compare with the solution in [84] that does not require prior knowledge. In order to take seller locality into consideration, it takes a conservative approach and requires every single seller-buyer pair to be budget balanced, and then finds the assignment for buyers one by one.

We randomly generate three regions and also randomly assign buyers to one of these regions with the same probability. District requires a pre-selected parameter that specifies the maximum number of winning buyers, and picks that many highest bid buyers and uses the bid of the first buyer that is not selected as the amount to charge to the winning buyers. We vary



(a) Efficiency (b) Revenue

(c) Utilization

Figure 10.5: Local sellers.

that parameter from 10 to 160.

Figure 10.5 summarizes the result. We make three observations: First, our scheme is significantly better in all the experiments. For example, DA^2 out-performs District by 2.5x to 12x in efficiency, 62% to 4.5x in revenue, and 4.5x to 20.5x in utilization. The main reason is that District requires each seller-buyer pair to be budget balanced. This artificial restriction makes a trade hard to take place. Our scheme not only benefits from considering all the buyers from all regions, but also uses cross-region budget balance to allow regions with higher revenue to help weaker regions, thus achieving much better performance. Second, the pre-selected parameter (*i.e.*, the maximum number

of winning buyers in District) has significant impact on the performance of District and the best parameter is difficult to determine. If the parameter is too small, too few buyers get selected; if the parameter is too big, the payment from winning buyers becomes too low and only few sellers have their budget balance requirement satisfied as a result. In comparison, our scheme does not require a pre-selected parameter. Third, compared with the default setting as in the previous evaluation where a seller can sell to all buyers, the absolute efficiency, revenue, and utilization are all lower. This is as what we would expect since the sellers here only sell in a specific region, which may not cover all buyers.

Chapter 11

Related Work

Our work is related to the following research areas:

Social networks: The explosive growth of online social networks has attracted significant attention [5, 11, 7, 57]. Previous studies (*e.g.*, [49, 50, 79]) show that path-ensemble based proximity measures, such as Katz measure [42], rooted PageRank [49, 50], and escape probability [79], are effective in predicting links between users. [76] develops proximity embedding algorithm for efficient and accurate proximity estimation in large social networks. It is applicable to all path-ensemble based proximity measures, so we use it in our study.

Mobile social networks: The popularity of mobile social networks has increased rapidly. [24] describes social serendipity to perform matchmaking in mobile social networks. Loopt[52] is a mobile geo-location service that notifies users of friends' location and activities via detailed interactive maps. Both schemes rely on a centralized server. Mobiclique[68] is a middleware that allows mobile phone users to connect to others over ad-hoc networks to exchange social network identity information and forward messages opportunistically. It assumes all users are trusted, and ignores privacy and security.

Privacy in online social networks: [12] applies attribute-based encryption to provide fine-grained access control to the personal information. [80] proposes a social attestation that certifies a social relationship between any two parties. The recipient of a social attestation can use it to prove to a third party (*e.g.*, an online system) that (s)he has a certain relationship with the sender and get access to the restricted content. These works are complementary to our protocols. [28] develops a cryptographic private matching technique that exploits friend-of-friend relationships to automatically generate whitelists for email senders, and [27] proposes enhancement. Since set intersection is a special case of dot product, our verifiable secure dot product protocol can be applied to their context to provide stronger security guarantees.

Privacy in wireless networks: Wireless network privacy becomes an increasing concern due to ease of eavesdropping. [65] shows that a number of explicit identifiers in 802.11 MAC header can be used to identify and track users. *SlyFi* [30] is an 802.11-like link layer protocol that obfuscates all transmitted bits to remove explicit identifiers and increase privacy. [20] provides flexible presence sharing between users with social relationships by broadcasting clique signals. It provides presence sharing among strangers by letting users broadcast opaque identifiers, which can be resolved to an identity at a centralized trusted broker. This is related to our virtual ID, but we use virtual ID as a public key to communicate directly with other users in the mobile social network.

Privacy-preserving computation: There are two classes of privacy-preserving

computation: encryption-based techniques and obfuscation-based techniques. Encryption-based protocols like [29] are based on homomorphic encryption and are computationally expensive. The obfuscation based schemes[38, 81] are light weight but tend to leak some information. Privacy-preserving dot product protocols are important in the area of distributed range query computing and association mining [29, 38]. A unique feature of our approach is that it is both privacy-preserving and verifiable.

Cellular offloading: The need to complement cellular networks with other forms of connectivity has been considered in the past. The authors in [13] conduct measurements in a vehicular testbed, and report that Wi-Fi is available for only 11% of the time and 3G is available for 87% of the time. Moreover, they find that 3G and Wi-Fi availability are negatively correlated, *e.g.*, Wi-Fi is available in 50% of the times that 3G is not available. Lee *et al.*, in [48] use daily mobility patterns of 100 iPhone users to measure the amount of data Wi-Fi can offload. They find that Wi-Fi can offload 65% of data traffic without any delay; if 1 hour or longer delay can be tolerated, the offload traffic increases further by 29%. Zhuo *et al.*, in [97] leverage VCG based auction mechanism to incentivize mobile users to wait until they come in contact with a Wi-Fi AP. Authors in [22] quantify city-wide WiFi offloading gain. They show that even a sparse Wi-Fi network improves performance. Different from the above existing works, our paper focuses on how to incentivize third party resource owners to offload cellular traffic. The work in [18] is closest to ours. It proposes a VCG reverse auction framework to buy femtocell resources. As

mentioned in Section 1, it does not address the three unique challenges we focus on, namely, diverse spatial coverage, traffic uncertainty, and collusion. The scheme is similar to the local allocation in spirit in that it statically determines the amount of third-party resource to buy for each region.

Spectrum auctions: There are a number of works on spectrum auctions. [95] is the pioneering work that proposed using double sided spectrum auctions. There have been several works on improving TRUST [89, 37, 91]. Although these works alleviate certain problems of TRUST, they are still similar in spirit: they still use random independent sets and enforce uniform price for buyers assigned the same channel. So they still suffer from low efficiency and poor fairness. [89] improves on TRUST to sacrifice fewer buyers. However, it uses a single sided auction, a much simpler problem. [37] builds on [89], but focuses on the privacy aspect instead of auction design. [91] proposes a new definition of group bid which achieves better auction performance and preserves all economic properties. So we also compare with [91] in our evaluation.

There are also some spectrum double auction designs that are not based on TRUST. However, many of them neglect the challenges in designing a truthful double auction and the need to capture a complicated conflict graph. [83] proposes an online double sided spectrum auction where buyers can come and request for resource at different times. However, they assume a complete conflict graph (*i.e.*, everyone interferes with everyone else), which simplifies the competition pattern and disables spectrum reuse. [90] proposes a single unit

double auction with discriminatory pricing, but their auction is not theoretically proven to be truthful. Some other works try to make double spectrum auction possible in more realistic settings. [25] proposes a double auction similar to TRUST, but takes buyers' frequency preference into consideration while forming groups. This is complementary to our work. [84] focuses on incorporating market locality. Our design also supports this feature and achieves much better performance as shown in Section 10. In addition, [39] proposes a double auction based approach for mobile data offloading, but it does not consider wireless interference as in spectrum auctions.

Chapter 12

Conclusion

In this work we focus on designing effective and mathematically sound approaches to facilitate collaborative mobile services. We first consider the collaboration between end users in a distributed mobile service: friend discovery in mobile social networks. This service requires two conflicting goals in collaboration simultaneously, information sharing (verifiability) and information protection (privacy). To address that challenge, we develop a secure friend discovery protocol that achieves both privacy and verifiability. Then to reduce the computation overhead we use a two phase protocol where in the first phase a light weight protocol that only ensures privacy is used, and the second phase is only triggered if the first phase is successful. Our protocols are the first verifiable secure dot product protocols and our solution is widely applicable to secure multiparty computations, privacy preserving data mining, etc.

Second, we study a instance of cross level collaboration and propose iDEAL to enable the cellular service provider to purchase and leverage third-party resources on demand through reverse auctions. iDEAL uses a truthful auction mechanism to incentivize true information sharing and avoid lying.

iDEAL also mitigates potential collusion between third party resource owners. Moreover iDEAL address several unique challenges in cellular offloading, such as diverse spatial coverage and dynamic traffic demands.

Finally we focus on collaboration at the service provider level. We propose DA^2, a novel double auction mechanism for cellular service providers to reallocation the spectrum resource in a dynamic fashsion. DA^2 improves on existing spectrum double auctions by adopting a new design approach that separates the buyer side design and the seller side design. By leveraging the clustered property of real world conflict graphs, DA^2 allocates spectrum resource more efficiently and achieves higher efficiency and revenue in the auction. We also carefully design the partition-and-merge framework in DA^2 such that all desired economic properties - truthfulness, budget balance and individual rationality - are preserved.

Appendix

Proof of Theorem 11:

Proof. To change N by lying, a seller i needs to change the point where the budget balance is satisfied, which requires the seller to change the total selling price. Let $S(N)$ be the total selling price when N channels sell and $R(N)$ be the total revenue. Let $U_i(x, N)$ be the utility of seller i when it asks x and N channels sell. Since the seller side design is truthful, we have $U_i(v_i, N) \geq U_i(x, N)$, where v_i is i's true valuation and x can be any value.

We consider the following cases:

(1) i is a winning seller: i cannot change N as long as it still wins. If it changes N and now loses, its utility does not improve.

(2) i is a losing seller:

(a) Lie by asking higher. Obviously, N cannot increase in this case, since $S(N)$ does not reduce for any N. i is a losing seller in the original case so it is not within the top N. i cannot enter the top N by asking even higher.

(b) Lie by asking lower. We only consider the case when i becomes a winner, because its utility is still 0 if it still loses. We then divide into the following cases.

Case 1: N does not change: i's utility cannot improve because $U_i(v_i, N) \geq U_i(x, N)$.

Case 2: N reduces to M: i is not in the top M when it bids truthfully, so $U_i(v_i, M) = 0$. Thus i either receive 0 utility or negative utility because

$U_i(v_i, M) \geq U_i(x, M)$

Case 3: N increases to M:

(i) i is in the top M when it bids truthfully, and the $M+1$-th asking price does not change. So $S(M)$ does not change. In the original case, only N channels sell, so we know $S(M) > R(M)$. So budget balance is not satisfied and this case cannot happen.

(ii) i is not in the top M sellers when it asks truthfully. So $U_i(v_i, M) = 0$. i's utility cannot increase in this case because $U_i(v_i, M) \geq U_i(x, M)$. □

Theorem 13. *Bid-independent merge is truthful when N is fixed.*

Proof. It is easy to see that the design is monotonic, *i.e.*, if a winner wins at v, it still wins if it bids $b > v$, as bidding higher does not reduce the group bid and all other decisions are bid-independent. Next we verify that the price we charge is the critical value. Specifically we charge a winner the first losing group bid in its subgraph divided by k, if k members of its group bid higher than that. We find the maximum k, so no more members can be admitted and still afford the fair share. To prove it is the critical value, we should show (1) a buyer still wins if it bids higher, and (2) a buyer loses if it bids lower. (1) holds because the buyer's group bid (which is greater than k times of the critical value) is still higher than the first losing group and the merge process is bid-independent. To see (2), we consider three cases: (i) k remains the same, (ii) k increases, and (iii) k decreases. For (i), if a buyer bids lower than that value, it cannot afford its fair share for the same k even if its group still wins.

(ii) cannot happen because other members' bids do not change and no new members can afford the fair share. In (iii), the fair share is even larger as k decreases, so the buyer does not win either. Thus a buyer never wins if it bids lower than the price we charge. Therefore the buyer side with bid-independent merge is truthful when N is fixed according to Theorem 9. □

Theorem 14. *Bid-independent merge based allocation does not allow a buyer to unilaterally change N and gain.*

Proof. The proof has same structure as the proof to Theorem 11 but from a buyer side. □

Theorem 15. *In our extension to incorporate seller locality, the price we pay to a seller according to our non-uniform seller side pricing is the critical value of that seller.*

Proof. Suppose seller i becomes budget balanced after dropping T sellers. To prove i is charged with its critical value, we show (1) i wins if it asks lower, and (2) i loses if it asks higher.

To prove (1): since i is not dropped in the original case and it now asks lower, it cannot be dropped before T sellers are dropped. So we consider two cases: (i) If i becomes budget balanced before T sellers are dropped, it wins. (ii) If i does not become budget balanced before T sellers are dropped, the same $T-1$ sellers are dropped as in the original case. That is because the order of their asking prices does not change, regions without i does not

change, and i's regions are still not budget balanced as in the original case. Thus the same T-th seller will be dropped, and i becomes budget balanced in the same region. So i wins.

To prove (2): similarly we consider (i) If i is dropped before T sellers are dropped, i does not win. (ii) If i is not dropped before T sellers are dropped, the same $T-1$ sellers are dropped as in the original case because all regions that have i are still not budget balanced and other regions are not affected by i's asking price. Now the scheme selects the highest asking seller that is not budget balanced to drop. In the original case, the T-th dropped seller, denoted as w, was selected. Now i asks higher than w, so i will be dropped instead and lose. □

Bibliography

[1] 3G Wi-Fi Seamless Offload.
http://www.qualcomm.com/documents/files/3g-wifi-seamless-offload.pdf.

[2] 802.11u. http://en.wikipedia.org/wiki/IEEE_802.11u.

[3] Hotspot 2.0.
http://www.networkworld.com/news/tech/2011/090711-80211u-hotspot-250556.html.

[4] 3GPP2 c.r1002-a cdma2000 evaluation methodology revision a, version 1.0. May 2009.

[5] L. A. Adamic, O. Buyukkokten, and E. Adar. A social network caught in the web. *First Monday*, 2003.

[6] N. Ahmed and S. Keshav. SMARTA: a self-managing architecture for thin access points. In *Proc. of CoNEXT*, 2006.

[7] Y.-Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of topological characteristics of huge online social networking services. In *Proc. of WWW*, 2007.

[8] Femto-cell range.
http://www.att.com/shop/wireless/devices/3gmicrocell.jsp?fbid=jZ_3W-pHS6d.

[9] Lawrence M Ausubel and Paul Milgrom. The lovely but lonely vickrey auction. *Combinatorial Auctions*, pages 1–37, 1961.

[10] Yoram Bachrach. Honor among thieves: collusion in multi-unit auctions. In *Proc. of AAMAS*, Richland, SC, 2010.

[11] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: Membership, growth, and evolution. In *Proc. of KDD*, 2006.

[12] Randy Baden, Adam Bender, Neil Spring, Bobby Bhattacharjee, and Daniel Starin. Persona: An online social network with user-defined privacy. In *Proc. of ACM SIGCOMM*, Aug. 2009.

[13] Aruna Balasubramanian, Ratul Mahajan, and Arun Venkataramani. Augmenting mobile 3G using WiFi. In *Proc. of MobiSys*, 2010.

[14] Robert M. Bell, Yehuda Koren, and Chris Volinsky. Chasing $1,000,000: How we won the Netflix Progress Prize. *Statistical Computing and Statistical Graphics Newsletter*, 18(2):4–12, 2007.

[15] J. C. Bezdek. Numerical taxonomy with fuzzy sets. *Journal of Mathematical Biology*, 1974.

[16] The code project: C# BigInteger class. `http://www.codeproject.com/KB/cs/biginteger.aspx`.

[17] Omer Biran and Francoise Forges. Core-stable rings in auctions with independent private values. CESifo Working Paper Series 3067, CESifo Group Munich, 2010.

[18] Yanjiao Chen, Jin Zhang, Qian Zhang, and Juncheng Jia. A reverse auction framework for access permission transaction to promote hybrid access in femtocell network. In *Proc. of IEEE INFOCOM*, 2012.

[19] Collectl tool. `http://collectl.sourceforge.net/`.

[20] Landon Cox, Angela Dalton, and Varun Marupadi. SmokeScreen: flexible privacy controls for presence sharing. In *Proc. of ACM MobiSys*, 2007.

[21] Peter Cramton. Competitive bidding behavior in uniform-price auction markets. *Hawaii International Conference on System Sciences*, 2, 2004.

[22] Savio Dimatteo, Pan Hui, Bo Han, and Victor O.K. Li. Cellular traffic offloading through wifi networks. In *Proc. of IEEE MASS*, 2011.

[23] Wei Dong, Zihui Ge, and Seungjoon Lee. 3G meets the internet: understanding the performance of hierarchical routing in 3G networks. In *Proc. of ITC*, 2011.

[24] Nathan Eagle and Alex Pentland. Social serendipity: Mobilizing social software. *IEEE Pervasive Computing*, Apr. 2005.

[25] Xiaojun Feng, Yanjiao Chen, Jin Zhang, Qian Zhang, and Bo Li. TAHES: a truthful double auction mechanism for heterogeneous spectrums. *Wireless Communications, IEEE Transactions on*, 11(11):4038 –4047, Nov. 2012.

[26] B. Fortz, J. Rexford, and M. Thorup. Traffic engineering with traditional IP routing protocols. *IEEE Comm. Magazine*, 40(10), Oct. 2002.

[27] Michael J. Freedman and Antonio Nicolosi. Efficient private techniques for verifying social proximity. In *Proc. of IPTPS*, Feb. 2007.

[28] S. Garriss, M. Kaminsky, M. J. Freedman, B. Karp, D. Mazieres, and H. Yu. RE: Reliable Email. In *Proc. of NSDI*, 2006.

[29] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On private scalar product computation for privacy-preserving data mining. In *Proc. of ICISC*, 2004.

[30] Ben Greenstein, Damon McCoy, Jeffrey Pang, Tadayoshi Kohno, Srinivasan Seshan, and David Wetherall. Improving wireless privacy with an identifier-free link layer protocol. In *Proc. of MobiSys*, 2008.

[31] Handover. `http://xa.yimg.com/kq/groups/19564168/454394754/name/05-Handover.pdf`.

[32] Handover tutorial. `http://www.radio-electronics.com/info/cellulartelecomms/gsm_technical/handove`

[33] Masaharu Hata. Empirical formula for propagation loss in land mobile radio services. *IEEE Trans. Vehicular Tech.*, VT-29(3):317–325, Aug. 1980.

[34] Shawndra Hill, Foster Provost, and Chris Volinsky. Network-based marketing: Identifying likely adopters via consumer networks. *Statistical Science*, 2006.

[35] Steven S. Hong, Jeffrey Mehlman, and Sachin Katti. Picasso: flexible RF and spectrum slicing. In *Proc. of ACM SIGCOMM*, 2012.

[36] Hostap. `http://hostap.epitest.fi/hostapd/`.

[37] Qianyi Huang, Yixin Tao, and Fan Wu. Spring: A strategy-proof and privacy preserving spectrum auction mechanism. In *IEEE INFOCOM*, 2013.

[38] Ioannis Ioannidis, Ananth Grama, and Mikhail Atallah. A secure protocol for computing dot-products in clustered and distributed environments. In *Proc. of ICPP*, 2002.

[39] George Iosifidis, Lin Gao, Jianwei Huang, and Leandros Tassiulas. An iterative double auction for mobile data offloading. *IEEE WiOpt*, 2013.

[40] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proc. of ACM MOBICOM*, 2003.

[41] Mohamed Kassab, Abdelfettah Belghith, Jean-Marie Bonnin, and Sahbi Sassi. Fast pre-authentication based on proactive key distribution for 802.11 infrastructure networks. In *Proc. of the 1st ACM workshop on Wireless multimedia networking and performance modeling*, 2005.

[42] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 1953.

[43] Florian Kerschbaum and Orestis Terzidis. Filtering for private collaborative benchmarking. In *Proc. of LNCS*, 2006.

[44] V. Krishna. *Auction theory*. Academic Press, 2002.

[45] Balachander Krishnamurthy and Craig E. Wills. On the leakage of personally identifiable information via online social networks. *ACM SIGCOMM Computer Communications Review*, Jan. 2010.

[46] Romain Kuntz and Jean Lorchat. Versatile IPv6 mobility deployment with dual stack mobile IPv6. In *Proc. of MobiArch*, 2008.

[47] Andrei Lapets and Alex Levin. Restricted truthful combinatorial auction mechanisms, 2007.

[48] Kyunghan Lee, Injong Rhee, Joohyun Lee, Song Chong, and Yung Yi. Mobile data offloading: how much can wifi deliver? In *Proc. of CoNEXT*, 2010.

[49] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proc. of CIKM*, 2003.

[50] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *J. Am. Soc. Inf. Sci. Technol.*, 2007.

[51] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129 – 137, 1982.

[52] Loopt. `http://www.loopt.com`.

[53] U. V. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 2007.

[54] R. Preston McAfee. A dominant strategy double auction. *Journal of Economic Theory*, 56(2):434–450, April 1992.

[55] Marina Meila and Jianbo Shi. A random walk view of image segmentation. In *Proc. of AI and STATISTICS (AISTATS)*, 2001.

[56] Arunesh Mishra Min-Ho, Min ho Shin, and William A. Arbaugh. Pro-active key distribution using neighbor graphs. *IEEE Wireless Comm.*, 2004.

[57] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *Proc. of IMC*, 2007.

[58] Mobile Broadband Review 2010.
http://mobile-broadband-services-review.toptenreviews.com/.

[59] Mobile social networking apps spark privacy concerns.
http://channel.hexus.net/content/item.php?item=25288.

[60] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Proc. of NIPS*, 2001.

[61] Tim Nieberg, Johann Hurink, and Walter Kern. A robust PTAS for maximum weight independent sets in unit disk graphs. In *Proc. of International Conference on Graph-Theoretic Concepts in Computer Science*. Springer-Verlag, 2004.

[62] J. Padhye, S. Agarwal, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *Proc. of ACM IMC*, 2005.

[63] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proc. of EUROCRYPT*, 1999.

[64] Paillier's homomorphic cryptosystem (java implementation).
http://www.csee.umbc.edu/~kunliu1/research/Paillier.html.

[65] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall. 802.11 user fingerprinting. In *Proc. of MobiCom*, 2007.

[66] Sang-Hee Park, Hye-Soo Kim, Chun-Su Park, Jae-Won Kim, and Sung-Jea Ko. Selective channel scanning for fast handoff in wireless LAN using neighbor graph. *Proc. of PWC*, 2004.

[67] Utpal Paul, Anand Kashyap, Ritesh Maheshwari, and Samir R. Das. Passive measurement of interference in WiFi networks with application in misbehavior detection. *IEEE Transactions on Mobile Computing*, 2012.

[68] Anna-Kaisa Pietiläinen, Earl Oliver, Jason LeBrun, George Varghese, and Christophe Diot. Mobiclique: middleware for mobile social networking. In *Proc. of WOSN*, 2009.

[69] Powertutor : A power monitor for android-based mobile platforms. `http://powertutor.org//`.

[70] Lili Qiu, Yang Richard Yang, Yin Zhang, and Scott Shenker. On selfish routing in internet-like environments. In *Proc. of ACM SIGCOMM*, Aug. 2003.

[71] Matthew Roughan, Mikkel Thorup, and Yin Zhang. Traffic engineering with estimated traffic matrices. In *Proc. of IMC*, Oct. 2003.

[72] M. Shafiq, L. Ji, and A. Liu. Characterizing and modeling Internet traffic dynamics of cellular devices. In *Proc. of ACM SIGMETRICS*, 2011.

[73] Srikant Sharma, Inho Baek, and Tzi-Cker Chiueh. Omnicon: a mobile ip-based vertical handoff system for wireless LAN and GPRS links. *Softw. Pract. Exper.*, 37, 2007.

[74] D.B. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming A*, 1993.

[75] Vivek Shrivastava, Shravan Rayanchu, Suman Banerjee, and Konstantina Papagiannaki. Pie in the sky: Online passive interference estimation for enterprise WLANs. In *Proc. of NSDI*, 2011.

[76] Han Hee Song, Tae Won Cho, Vacha Dave, Yin Zhang, and Lili Qiu. Scalable proximity estimation and link prediction in online social networks. In *Proc. of IMC*, 2009.

[77] Sunk costs. `http://en.wikipedia.org/wiki/Sunk_costs`.

[78] Kun Tan, Haichen Shen, Jiansong Zhang, and Yongguang Zhang. Enable flexible spectrum access with spectrum virtualization. In *Proc. of Dyspan*, Sept. 2012.

[79] Hanghang Tong, Christos Faloutsos, and Yehuda Koren. Fast direction-aware proximity for graph mining. In *Proc. of KDD*, 2007.

[80] A. Tootoonchian, S. Saroiu, Y. Ganjali, and A. Wolman. Lockr: Better privacy for social networks. In *Proc. of CoNext*, Dec. 2009.

[81] Jaideep Vaidya and Chris Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proc. of ACM SIGKDD*, 2002.

[82] W. Vickrey. Counterspeculation, Auctions and Competitive Sealed Tenders. *Journal of Finance*, pages 8–37, 1961.

[83] ShiGuang Wang, Ping Xu, XiaoHua Xu, ShaoJie Tang, XiangYang Li, and Xin Liu. TODA: truthful online double auction for spectrum allocation in wireless networks. In *Proc. of New Frontiers in Dynamic Spectrum*, pages 1 –10, Apr. 2010.

[84] Wei Wang, Baochun Li, and Ben Liang. District: Embracing local markets in truthful spectrum double auctions. In *Proc. of IEEE SECON*, 2011.

[85] Wikipedia. Social network. `http://en.wikipedia.org/wiki/Social_network`.

[86] Wireless Geographic Logging Engine. http://wigle.net.

[87] Worldwide pricelist for iPhone 3G Plans. `http://www.unwiredview.com/2008/07/16/worldwide-pricelist-for-iphone-3g-plans`

[88] WPA Supplicant. `http://hostap.epitest.fi/wpa_supplicant/`.

[89] Fan Wu and N. Vaidya. Small: A strategy-proof mechanism for radio spectrum allocation. In *Proc. of IEEE INFOCOM*, pages 81 –85, april 2011.

[90] Liyao Xiang, Gaofei Sun, Jing Liu, Xinbing Wang, and Li Li. A discriminatory pricing double auction for spectrum allocation. In *Proc. of WCNC*, Apr. 2012.

[91] Enxin Yao, Luxi Lu, and Wei Jiang. An efficient truthful double spectrum auction design for dynamic spectrum access. In *Proc. of CROWNCOM*, Jun. 2011.

[92] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. SybilGuard: Defending against Sybil attacks via social networks. In *Proc. of SIGCOMM*, 2006.

[93] Sangki Yun, Daehyeok Kim, and Lili Qiu. Fine-grained spectrum adaptation in wifi networks. In *Proc. of ACM MobiCom*, Sept. 2013.

[94] Xia Zhou, Zengbin Zhang, Gang Wang, Xiaoxiao Yu, Ben Y. Zhao, and Haitao Zheng. Practical conflict graphs for dynamic spectrum distribution. In *Proceedings of the ACM SIGMETRICS*, pages 5–16, New York, NY, USA, 2013. ACM.

[95] Xia Zhou and Heather Zheng. Trust: A general framework for truthful double spectrum auctions. In *Proc. of IEEE INFOCOM*, 2009.

[96] Zhichao Zhu, Guohong Cao, Ram Keralapura, and Antonio Nucci. Characterizing data services in a 3G network: Usage, mobility and access issues. In *Proc. of ICC*, 2011.

[97] Xuejun Zhuo, Wei Gao, Guohong Cao, and Yiqi Dai. Win-coupon: An incentive framework for 3G traffic offloading. In *Proc. of ICNP*, 2011.